1 Abstract

Influencer marketing has many complex moving parts, and every stakeholder is paying the price. Content creators, talent managers, agencies, and brand teams are trapped in a fragmented ecosystem where operational inefficiencies cripple growth. SocialLens solves this by using a layered, multi-agent orchestration system that automates analytics, compliance, and reporting for creators and influencer managers, as well as our own MCP Server that consolidates, automates and delivers provenance-rich, compliance-ready analytics.

SocialLens empowers creators to scale sustainably, managers to operate efficiently, and brands to trust their campaigns, thus unlocking a new standard for influencer marketing intelligence.

1.1 Problem

Content creators, especially the solopreneurs with audiences ranging from 50k to 500k followers, face the problem of transition to scale (Yap, 2025). As much as they are large enough to demand professional-grade operations, these content creators remain too small to afford the infrastructure or a social media manager to support them. Content creators are not merely producing content; they are running micro-enterprises that span multiple disciplines: media production, audience analytics, customer service, brand management, legal compliance, and financial administration (Patrecia Meliana, 2025). On a daily basis, content creators forfeit up to 10 hours on repetitive, non-creative tasks, such as managing a chaotic ecosystem of browser tabs, spreadsheets, monitor performance metrics across a few platforms, message inboxes and ever changing platform policies. They also have to fork out time and effort to deal with the inevitable demonetisation and reputational damage. Yet hiring help remains financially out of reach, and automation tools are often fragmented, expensive, or poorly integrated. These creators are victims of their own success - overwhelmed by operational complexity, paralysed by risk, and unable to scale sustainably. On the other hand, talent managers handle multiple creators and need comparable, consolidated reports across platforms. Current workflows involve manual data collection from disparate APIs, leading to delays and inconsistencies. From a B2B standpoint, agencies preparing sponsor reports before, during, and after campaigns face data silos and format mismatches, making it hard to deliver timely, audit-ready insights to clients (Ishola, 2025). Brand teams require secure, reliable access to influencer performance data for compliance and campaign evaluation, but existing solutions lack provenance, transparency, and controlled access. SocialLens is our all-in-one solution to help stakeholders in influencer markets, such as content creators, talent managers, agencies as well as brand teams manage and optimise these repetitive, non-creative tasks.

For a detailed market analysis and competitor analysis, see Appendix A.

1.2 Who are our stakeholders

Content Creators are incentivised to use our platform when they are operationally overloaded, face the lack of operational efficiency. They want performance tracking, compliance support, and monetisation insights from us. They can also be early-stage users who need personalised recommendations and growth strategies, and have the potential to convert into paying customers as they scale.

Talent Managers handle multiple influencers across platforms and have to deal with data silos and inconsistent formats. SocialLens normalises metrics into a consistent schema.

Agencies have to prepare sponsor reports before, during, and after campaigns which takes time and is prone to errors due to inconsistent data formats across platforms. The absence of audit-ready provenance further complicates compliance, exposing agencies to reputational and legal risks. These inefficiencies consume considerable time during the campaign lifecycle, especially in regulated markets. By automating tool execution and data aggregation, transforming manual workflows into streamlined processes.

Brand Teams have to monitor influencer performance securely and reliably, and the complexity of managing multi-market campaigns amplifies these risks, making transparency and controlled access critical for global brands. By delivering provenance-rich analytics and enforcing strict governance, we empower brand teams to make informed decisions confidently.

2.1 General Solution Architecture

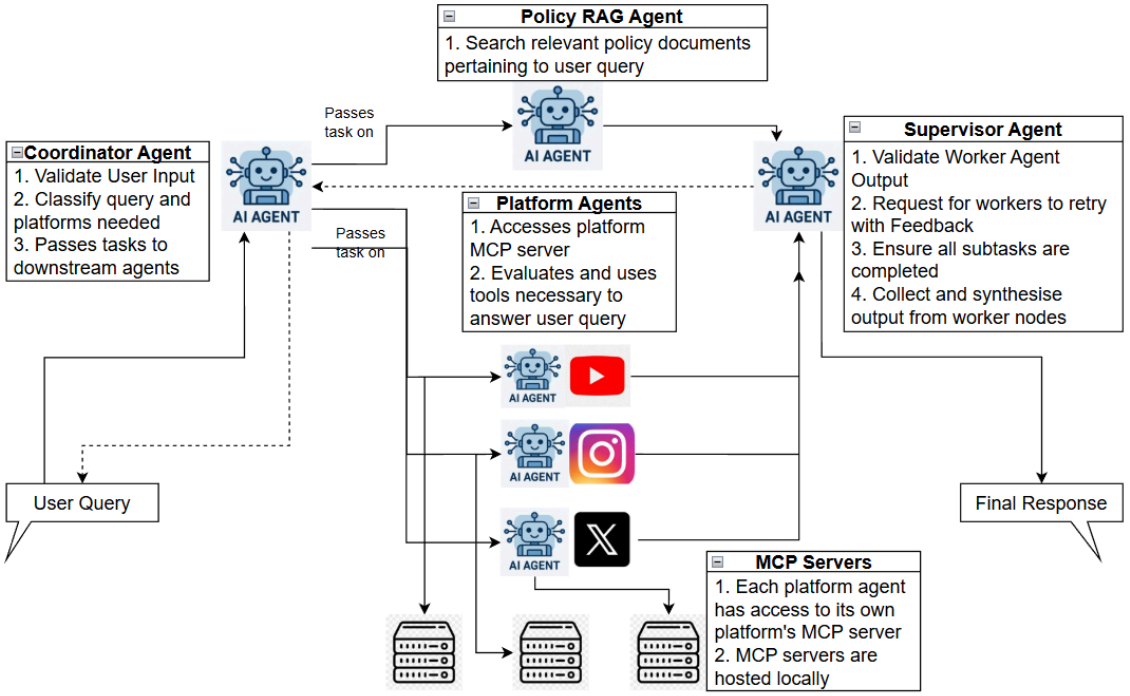


Figure 1. Overall Multi Agent Architecture

2.1.1 Multi Agent Architecture Considerations

The system is architected as a **hierarchical multi-agent system** (MAS) to ensure a robust separation of concerns and to promote modularity. This architecture is implemented with **LangGraph** as an orchestrator. This approach integrates two primary agentic patterns: a **Supervisor-Worker pattern** for task decomposition and a **Critic-Loop pattern** for quality assurance (Yeo, 2025). This design deliberately separates the initial task *routing* logic (handled by the Coordinator) from the final *validation* and *synthesis* logic (handled by the Supervisor), creating a more resilient and maintainable system. A detailed System architecture is outlined in the appendix for brevity.

Manager-Worker Design

The core of our task execution framework relies on the **Supervisor-Worker pattern**. In this design, the **Coordinator Agent** serves as the single "Manager". It functions as the system's primary entry point to delegate sub tasks to the appropriate "Worker" agents. The worker layer consists of specialized, independent agents, chiefly the **Policy RAG Agent** for compliance-related document retrieval and the **Platform Agents** for executing specific tool calls against their respective MCP servers.

Overall the design prioritises extensibility, modularity and maintainability, as new worker agents can be added to handle new tasks without altering the manager's core routing logic.

2.2 Coordinator Agent

2.2.1 Coordinator Agent as query orchestrator

Content creators ask diverse questions ranging from platform policies, tax compliance to engagement metrics, often mixing different types of queries. Without intelligent routing, all queries would use all agents, wasting resources and producing non-contextual responses.

2.2.2 Coordinator Agent Architecture

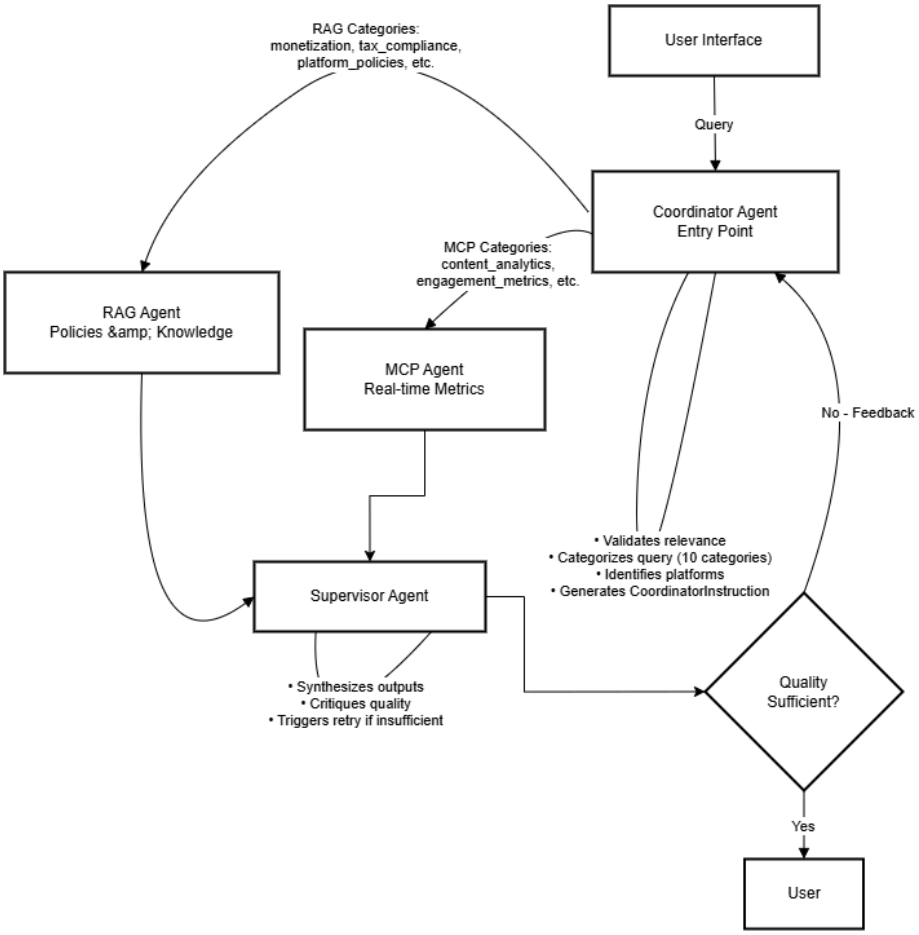


Figure 2. Coordinator Agent Architecture

The CoordinatorAgent serves as the entry point and orchestrator for the content creator assistant's multi-agent system. It has three roles - a filter, a router and a validator. The filtering ensures that only relevant queries reach downstream agents. The router will provide structured instructions that are catered to each agent's capabilities, such as categories and platforms for RAG to scope the database search, and platform, metrics and time range for the platform MCP agents. The CoordinatorAgent will also validate that the required credentials are available before processing the query.

2.2.3 Coordinator Agent Process Flow

When a user query comes in, eg. "Am I eligible for YouTube monetization?", the CoordinatorAgent will first validate if the query is a relevant question to SocialLens. Then, it will categorise the query according to the pre-defined RAG/MCP categories, as well as identify the platform needed (Youtube) for the query. It will then decide the route needed (MCP/RAG, or both). It will then pass the `rag_filters` to the RAG agent, or the `mcp_requirements` to the MCP agents.

2.2.4 Coordinator Agent Testing

Unit tests were carried out on the query validation/error handling of the CoordinatorAgent, as well as the platform categorisation, and routing logic. The CoordinatorAgent achieved 100% accuracy on manually created question answer sets (10 irrelevant, 10 relevant). An example question-answer set would be in JSON: {question: “Am I eligible for YouTube monetisation?”, “answer”: “query validation”: “true”, “platform categorisation”: “YouTube”, “routing logic”: “MCP, RAG”}.

2.2.5 Coordinator Agent Evaluation

Given more time, some features that we would have wanted to add were to implement query caching for repeated questions, as well as multi-modal support for queries. Some features to improve user experience would be to also have proactive query suggestions based on past user questions, and also to fine-tune a smaller model specifically for the task of validation, categorisation and routing.

2.3 RAG Agent

The RAG (Retrieval-Augmented Generation) agent is designed to function as a specialized compliance expert, mitigating the risks of non-compliance with dynamic platform policies. This is a critical function that standard, static-knowledge chatbots cannot fulfill. Their use creates significant risk, potentially leading users to face account suspension, demonetization, or loss of income due to acting on outdated information.

- **Factual Grounding:** Strict prompt engineering to ensure answers are fact-based.
- **Timeliness:** Regular scraping to ensure knowledge base is constantly updated
- **Niche Specificity:** Enabling the agent to find platform specific answers (e.g., "YouTube Partner Program") that general LLMs are not trained on.
- **Targeted Retrieval:** Enhanced search with a hybrid search strategy.
- **Trustworthiness:** Formatting answers with cited source URLS for credibility

2.3.1 RAG Agent Architecture Diagram

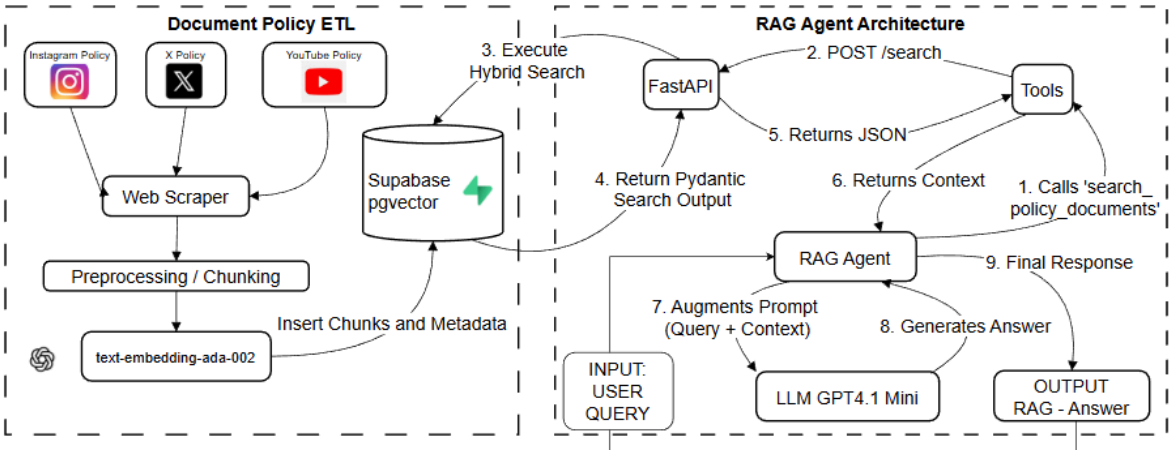


Figure 3. End-to-End RAG Architecture Diagram

The RAG Agent can be broken into two main workflows, **the data ingestion layer**, which manages the ETL process and the RAG Agent layer that handles **document retrieval and answer generation**.

2.3.2 RAG Agent Process Flow

Data Ingestion Layer (Document Policy ETL)

The offline Data Ingestion (ETL) pipeline is responsible for building and maintaining the system's knowledge base. It employs Playwright and Trafilatura to scrape and extract clean article text from the policy centers of YouTube, Instagram, and X. We utilised a recursive **chunking strategy** to preserve hierarchical and natural

paragraphing boundaries using LangChain's `RecursiveTextSplitter`. The text is then segmented into 1000-character chunks with a 200-character overlap to preserve semantic context. Each chunk is subsequently converted into a high-dimensional vector using OpenAI's `text-embedding-ada-002` model and loaded into a Supabase (PostgreSQL) database.

We chose Supabase as it provides a scalable, hosted and **serverless PostgreSQL database** with the `pgvector` extension, enabling a powerful combination of relational pre-filtering and efficient vector similarity search within a single, managed platform. As the current size of data is still small, a **flat index** is used to maximise accuracy, with the intention of using either a **HNSW index** or **IVF-Flat index** in the future when database size scales. Detailed technical design choices and detailed database schema are detailed in the appendix C.

Retrieval Augmented Generation Layer (RAG Agent Architecture)

The online RAG Agent handles live user queries via a FastAPI-served API. The database is exposed through this API to facilitate querying. When a query is received, the system executes a sophisticated hybrid search strategy within the Supabase instance. This search combines a `pgvector`-based semantic search with a `tsvector`-based keyword search. The results from both search modalities are aggregated and re-ranked using Ranked Reciprocal Fusion (RRF) to produce a single, highly relevant list of text chunks. These top-k chunks are then injected as context into a prompt for an LLM, which generates a factually-grounded, natural-language answer complete with source URL citations. Detailed implementation of hybrid search and RRF can be found in appendix C.

2.3.3 RAG Agent Evaluation

We quantitatively assessed our Retrieval-Augmented Generation (RAG) system using the industry standard **RAGAs** framework, evaluating both the retrieval and answer generation components.

To measure the efficacy of the retrieval stage, we employed a suite of **non-LLM-based** metrics which are *document accuracy, precision@k, recall@k, mean reciprocal rank (MRR)* These metrics provide objective, reproducible scores by comparing the document-IDs of retrieved items against a known ground truth.

The quality of the final, user-facing answer was assessed using **LLM-as-judge** metrics, which evaluate complex semantic qualities of the text like *Faithfulness* and *Answer Relevancy.* The RAGAs evaluation library was utilised for these metrics.

Experimental Procedure

The evaluation was executed through the following systematic, multi-step process:

1. **Test Set Curation:** A ground-truth dataset was established by curating source documents and applying recursive chunking.
2. **Synthetic Query Generation:** We employed an "LLM-as-author" methodology to create test queries, each intentionally mapped to a specific chunk to test single-hop retrieval.
3. **Evaluation Execution:** Each synthetic query was processed by the RAG system's retrieval component. The set of retrieved document-IDs was then compared against the ground-truth document-ID to compute the retrieval metrics. Subsequently, the generated answer was evaluated against the query and the source chunk to compute the generation metrics.
4. **Metric Aggregation:** Scores for each metric were aggregated across all test cases to provide a consolidated, quantitative performance summary.

The RAG agent demonstrated strong overall performance, excelling in both retrieval and generation for single-chunk queries. High scores in document accuracy and recall confirm its proficiency at retrieving pertinent information, while strong faithfulness and relevancy scores show its capability to generate accurate, well-grounded answers. For a detailed analysis, view appendix C.

2.3.4 RAG Agent Evaluation and Future Work

Overall, our RAG workflow was designed with scalability, performance, and responsiveness in mind. This strong architectural foundation not only meets our current objectives but also provides a clear path for future enhancements. To build upon this, we have prioritized the following areas to further advance the system's robustness, accuracy, and scale:

1. **Two-Stage Retrieval with Reranking:** The current system employs a single-pass retrieval, returning the top-*k* documents directly. A significant improvement would be to implement a two-stage process. This would involve an initial, fast retrieval followed by reranking stage utilising a more powerful model, such as a cross-encoder, to re-rank the top candidates which can significantly improve final document accuracy.
2. **Advanced Test Case Generation (Multi-Hop Retrieval):** Our current evaluation is limited to single-hop queries. To address more general and complex user questions, future work must expand test case generation to include multi-hop reasoning. The current RAG implementation could also be extended to a Graph RAG approach that leverages knowledge graphs to establish and traverse relationships between document chunks, enabling more sophisticated reasoning.
3. **Hybrid Retrieval with BM25:** The current agent's keyword matching relies on `tsvector`, a basic implementation. An enhancement would be to utilise BM25 (Best Match 25), the industry standard for lexical information retrieval to provide more comprehensive and accurate results.
4. **Real-Time Data Ingestion Pipeline:** The data ingestion pipeline is currently planned for periodic batch updates on a weekly basis. A more advanced implementation would involve creating a real-time, event-driven pipeline. By utilizing webhooks or other trigger-based mechanisms, the system could automatically detect, scrape, and index new or updated data as soon as it becomes available, ensuring the agent's knowledge base is perpetually up-to-date.

2.4 MCP Server

2.4.1 Business Value of MCP Orchestration and Gateway

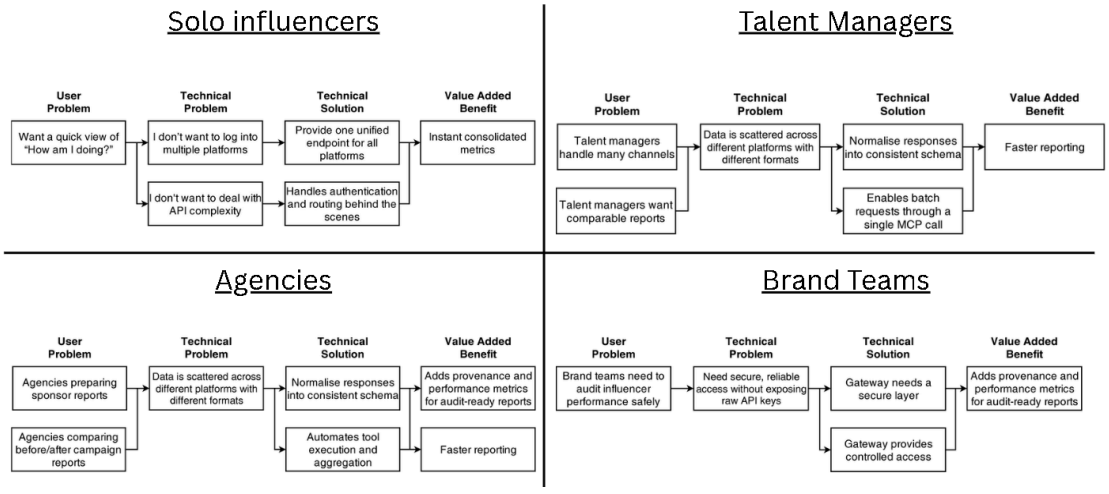


Figure 4: Business value of MCP Orchestration and Gateway to each stakeholder.

2.4.2 MCP Orchestrator Architecture Diagram

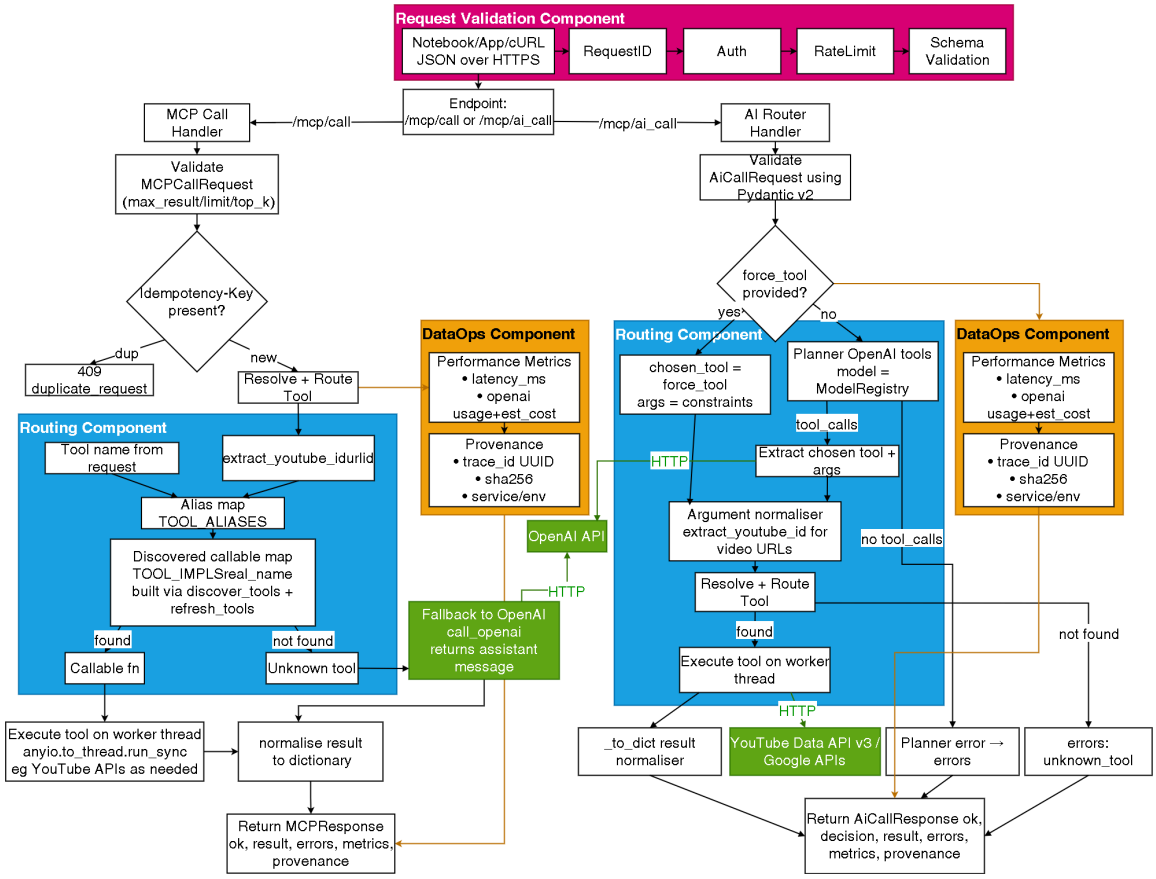


Figure 5: End-to-end flow of MCP server handling AI tool calling, validation, routing, and governance.

The MCP orchestration layer turns a set of heterogeneous social-platform tools into a single, governed workflow. Every request first passes through the validation component, which assigns a request ID, enforces API-key authentication and rate limits, and validates the JSON body against strict schemas. This guarantees that only authorised clients can invoke tools, protects upstream quotas, and ensures that bad inputs fail fast with traceable errors rather than silently corrupting reports. Once validated, the AI planner in the routing component interprets the user's natural-language query, selects the appropriate MCP tool and arguments, and normalises results into a consistent envelope. This orchestration hides the complexity of X, YouTube, and Instagram APIs, so solo influencers and talent managers can ask questions in everyday language while still getting structured, comparable metrics back. In parallel, the DataOps component instruments every step of the orchestration with latency, token usage, estimated model cost, and cryptographic provenance. These metrics are attached to the response, turning each call into an audit-ready record that agencies and brand teams can rely on when preparing sponsor reports or defending campaign performance. The net business value is faster reporting, lower engineering overhead, and higher trust: brands get secure, controlled access; agencies get defensible numbers with full lineage; and creators get a single, intelligent "performance endpoint" instead of juggling multiple platforms.

Additional details about MCP Orchestration and Gateway can be found in Appendix D.

2.4.3 MCP Gateway

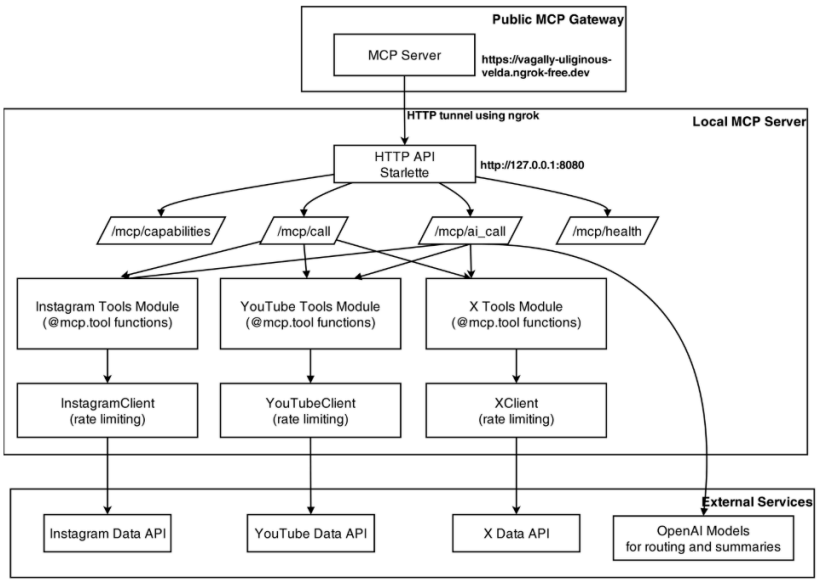


Figure 6: End-to-end flow of a MCP server handling AI tool calling, validation, routing, and governance.

The MCP gateway exposes the local MCP server as a single secure HTTPS endpoint, giving all four customer types a clean technical boundary and abstraction over the messy social and AI APIs. Solo influencers only need one URL and API key to hit /mcp/ai_call or /mcp/call, so their notebooks or low-code tools can run analytics from anywhere without dealing with OAuth, SDKs, or local networking. Talent managers get a stable integration point for dashboards and CRMs: all roster-wide analytics talk to the same gateway regardless of how many underlying tools modules (X, YouTube, Instagram) are added or changed, which makes feature roll-outs far easier. Agencies can plug ETL jobs, Airflow DAGs, and BI tools into this single endpoint and let the gateway plus local server handle tunnelling, rate limiting, and platform orchestration. The combination gives partners controlled, auditable access to influencer performance data without exposing sensitive infrastructure.

2.4.4 MCP Server Deployment

What each step does	Deployment steps
Start Local MCP Server	python deploy_mcp_servers\combined_mcp_server_v2.py
Deploy MCP Gateway	ngrok http INTERNAL ADDRESS
Query MCP Server on cmd / API testing tools (e.g. Postman) / python	curl -i HOSTNAME/mcp/ai_call -X POST -H "x-api-key:devkey123" -d "{\"id\": \"x\",\"query\": \"YOUR QUERY\"}"

2.5 Supervisor Agent

The Supervisor Agent functions as the system's **primary manager** and **quality controller**. Its overarching role is to oversee the entire workflow, ensuring that all responses are complete, accurate, and unified before being delivered to the user. It manages queries that require input from multiple agents, like the RAG and MCP agents, and performs a final quality check to ensure the output is sufficient.

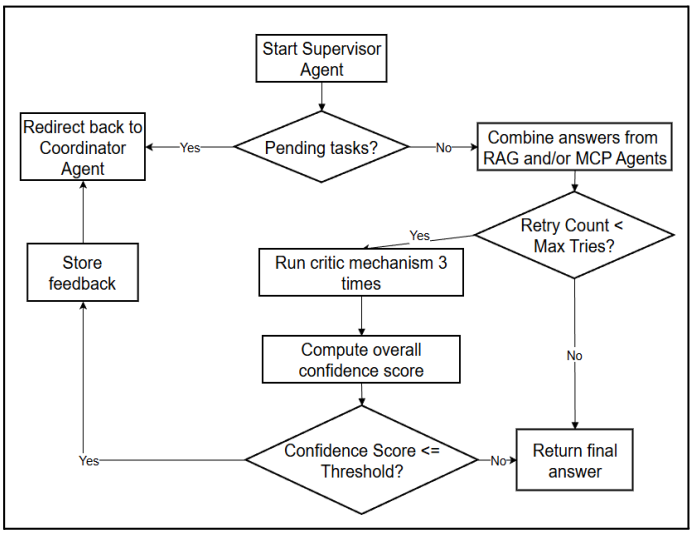


Figure 7: Workflow and Orchestration of Supervisor Agent

This agent's core tasks involve:

- **Task Completion Verification:** Checks if all required worker agents have finished their tasks.
- **Task Redirection:** If a task is incomplete, it routes the query back to the Coordinator Agent for re-assignment to the correct worker
- **Quality Critique:** Evaluates the generated answer step-by-step against a predefined quality threshold.
- **Feedback Generation & Revision:** If an answer is insufficient, it generates feedback. It then routes the query and the new feedback back to the Coordinator to trigger a retry by the worker agents
- **Answer Synthesis:** Once all tasks are complete and have passed the quality critique, it combines all outputs into a single, unified response.
- **Final Delivery:** Delivers the final, synthesized, and validated answer to the user.

2.5.1 Supervisor Agent Critique Scoring Mechanism

The critique mechanism of the Supervisor Agent is based on a self-probing Chain of Thought (COT) prompting strategy. The quality of the generated answers was evaluated using the three criteria defined in the table below.

Criteria	Guiding Question	Scoring Guideline
Relevance	Does the answer directly	• **0.0-0.3:** Mostly off-topic or only marginally

	address the user's question?	addresses ● **0.4-0.7:** Partially addresses the question but misses the main focus ● **0.8-1.0:** Highly relevant, directly addresses the question
Completeness	Does the answer cover all necessary aspects?	● **0.0-0.3:** Very incomplete; major parts are missing ● **0.4-0.7:** Partially complete; some key aspects are missing ● **0.8-1.0:** Comprehensive; covers all necessary points
Accuracy	Is the information plausible and consistent with facts or logic?	● **0.0-0.3:** Factually incorrect or logically flawed ● **0.4-0.7:** Mostly accurate; minor errors or inconsistencies ● **0.8-1.0:** Fully accurate and consistent with facts and logic

Table 1: Scoring Rubrics for Supervisor

The critique mechanism is designed to be rigorous and reproducible. The LLM evaluates the output and returns a JSON object containing a single-line actionable feedback sentence and scores for several key aspects. These scores are averaged with an equal 1:1:1 weighting to produce a single confidence score using the following equation.

$$Confidence\ Score\ = \frac{1}{3}Score_{relevance} + \frac{1}{3}Score_{Completeness} + \frac{1}{3}Score_{accuracy}$$

To mitigate the risk of arbitrary LLM judgments and ensure a stable evaluation, the entire critique process is executed three times. The confidence scores from all trials are then averaged to produce a final, aggregated score. This score is compared against a **predefined quality threshold of 0.8** to determine if the output is sufficient.

This multi-trial approach creates a deterministic and reproducible validation process, making the evaluation less susceptible to the non-determinism or hallucinations of a single LLM pass. The result is a credible and consistently high-quality output from the supervisor agent.

2.5.2 Supervisor Agent Testing and Evaluation

The current evaluation of the Supervisor Agent is centered on User Acceptance Testing (UAT) using predefined scenarios. These scenarios are derived from our core user stories and are designed to validate the agent's end-to-end logical workflow as depicted in its flowchart.

Based on some initial testing the current use of a 0.8 confidence threshold and using the 1:1:1 weight ratio for confidence scoring was able to produce satisfactory outcomes based on human evaluation. Detailed Example can be found in Appendix E

Future Roadmap for Robust Testing
For further testing in the future we plan to create a "golden set" benchmark of human-scored answers and decision flows. This will enable a formal meta-evaluation of the critique mechanism itself, allowing us to measure the statistical correlation between the agent's confidence scores and human quality judgments to rigorously validate its reliability as a judge. This process will enable formal data-driven optimization of the critique mechanism and robustness in future evaluation

2.6 Front End User Interface

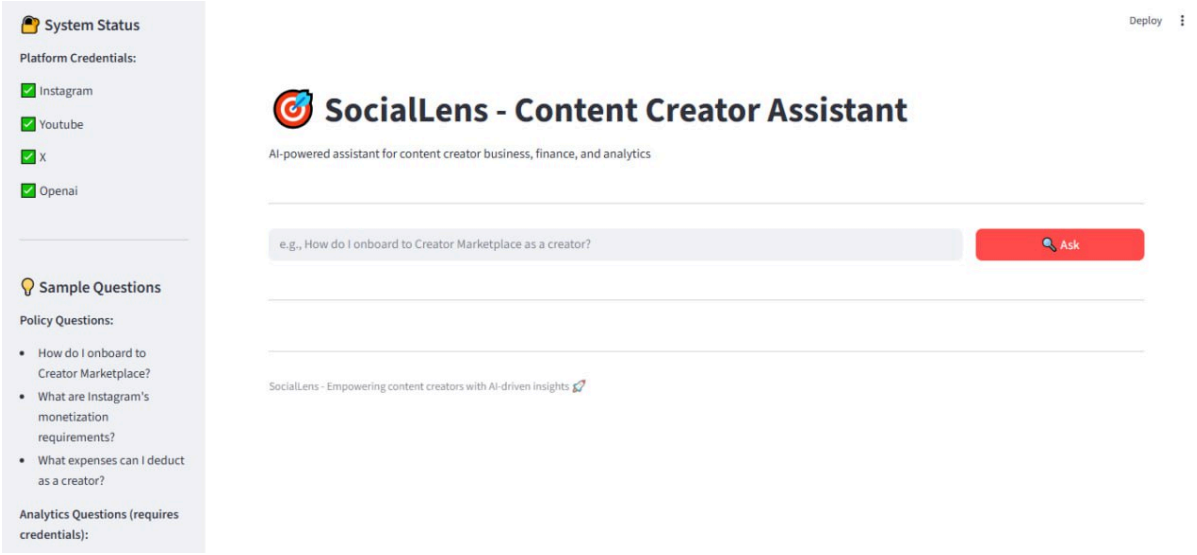


Figure 8: Landing Page of Application

For the UI layer, we chose Streamlit due to its effectiveness in rapid prototyping. This framework allowed us to quickly build a functional and intuitive front-end interface, centered around a simple text input for user queries.

3 Overall Limitations and Mitigation Strategies

The current prototype's primary limitation is its dependency on external platforms. The system's functionality is confined to Instagram, X, and YouTube , and its effectiveness is contingent upon the stability of third-party APIs and the quality of the RAG-generated content. This creates a significant risk regarding data freshness, as platform policies can change overnight, potentially rendering the RAG's advice outdated.

Our proposed short-term solution is to orchestrate scheduled jobs for data scraping. A more robust, long-term strategy involves implementing an event-driven system with webhooks to detect policy changes and trigger real-time data re-ingestion. If we are able to secure a partnership with the platform and update our policies, it would also mitigate this issue as we could achieve real time updates.

Usability and scalability present a second major challenge. The prototype requires users to manually generate their own developer API keys, a significant barrier for non-technical creators. This reliance on individual keys also introduces a scalability risk, as the MCP Agent is constrained by third-party API rate limits. The immediate plan is to provide a detailed user guide for key generation and implement intelligent caching for MCP API calls. The long-term solution is to build a secure OAuth 2.0 flow, enabling users to "Log in with Google/Instagram" to centralize and manage credentials securely and efficiently.

Finally, the system has functional limitations in retrieval and presentation. The RAG's Hybrid Search struggles with ambiguous queries (e.g., "Is this specific song okay?"), potentially returning a general policy that a user could misinterpret as a specific "yes," leading to copyright strikes. The short-term fix is to program the RAG Agent to explicitly state when it can only provide general advice. In the long term, a classifier will be trained to detect unanswerable queries, and the system may integrate new tools to verify specific licensing restrictions.

Furthermore, the current analytics output is text-based, which is ill-suited for visual-centric creators. This will be addressed initially by using Streamlit or Plotly for simple charts, with the ultimate goal of developing a fully custom, interactive front-end dashboard.

3.1 Future Roadmap

Building on these mitigation strategies, the future roadmap focuses on expansion and refinement. The first priority is to expand platform coverage to include other major social platforms, such as TikTok and LinkedIn. Concurrently, the RAG system's knowledge base will be enhanced to incorporate not only platform-specific policies but also broader governance and regulatory guidelines. Finally, to ensure the solution meets market needs, formal User Acceptance Testing (UAT) will be conducted with real content creators. This feedback will be crucial for identifying and prioritizing areas for enhancement in future iterations.

3.2 Conclusion

Scaling your influence should not mean scaling your stress. SocialLens transforms the chaos of manual workflows into clarity and control, giving creators the freedom to focus on creativity while managers gain the transparency they need to deliver trusted results. With AI-powered automation, unified dashboards, and secure multi-platform access, SocialLens cuts operational overhead, saving you hours every week. For just $25/month, you can centralise your analytics, simplify compliance, and unlock growth without the friction. Subscribe to us today. Let SocialLens handle the complexity so you can handle the creativity.

Appendix A: Business Standpoint

To understand the potential and competitive advantage of our proposed solution, we analyze it through the SWOT framework.

Strengths

- Insights and recommendations tailored based on every social media creator's internet persona
- Unified platform for analytics across multiple social media channels
- Time-saving automation for performance summaries and compliance checks

Weaknesses

- Heavy reliance on APIs and data access permissions from social media platforms
- Without prior knowledge or assistance, potential users may have difficulty generating keys and secrets from their own social media platforms
- Continuous need to update terms and conditions that will be used for RAG

Opportunities

- Growing creator economy, especially among younger demographics, not only in Singapore but across Southeast Asia
- Potential B2B expansion through partnerships with marketing agencies

Threats

- Changes in platform API policies or data-sharing restrictions
- Privacy concerns and data security regulations across different countries, such as the PDPA in Singapore

Feasibility of solution

From various perspectives, the feasibility of the solution can be evaluated through its technical, economic, and legal aspects.

1. **Technical Feasibility**
   Recent advancements in generative AI make it possible to generate proper insights as long as enough context and queries are provided. Data integration across multiple platforms can be achieved through MCPs, while compliance can be ensured with the help of RAG.

2. **Economic Feasibility**
   The creator economy in Singapore and globally continues to expand, attracting investments from agencies and creators themselves. Developing this platform requires a moderate initial investment but offers strong potential for revenue through various subscription models and premium features.

3. **Legal and Feasibility**
   The solution will comply with each platform's terms of service and local data privacy regulations, such as the PDPA in Singapore. By scraping sufficient context and conducting iterative improvements for RAG, compliance can be effectively maintained.

Market Fit Analysis

Market Sizing

Total Addressable Market (TAM)

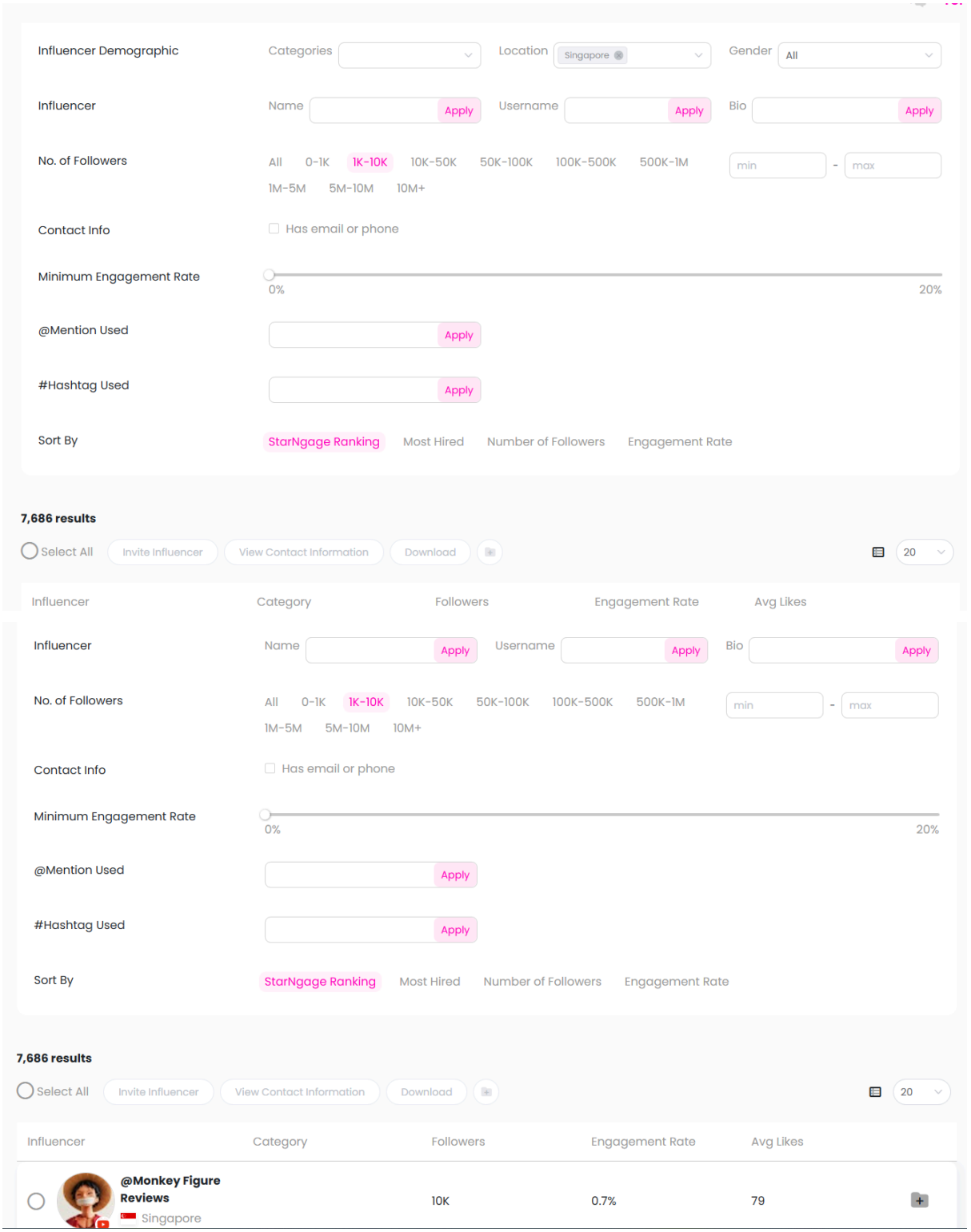
Our long-term opportunity is framed by the rapidly expanding Global Influencer Marketing Platform market (Grand View Research, 2024). As brands increasingly shift ad spend from traditional channels to

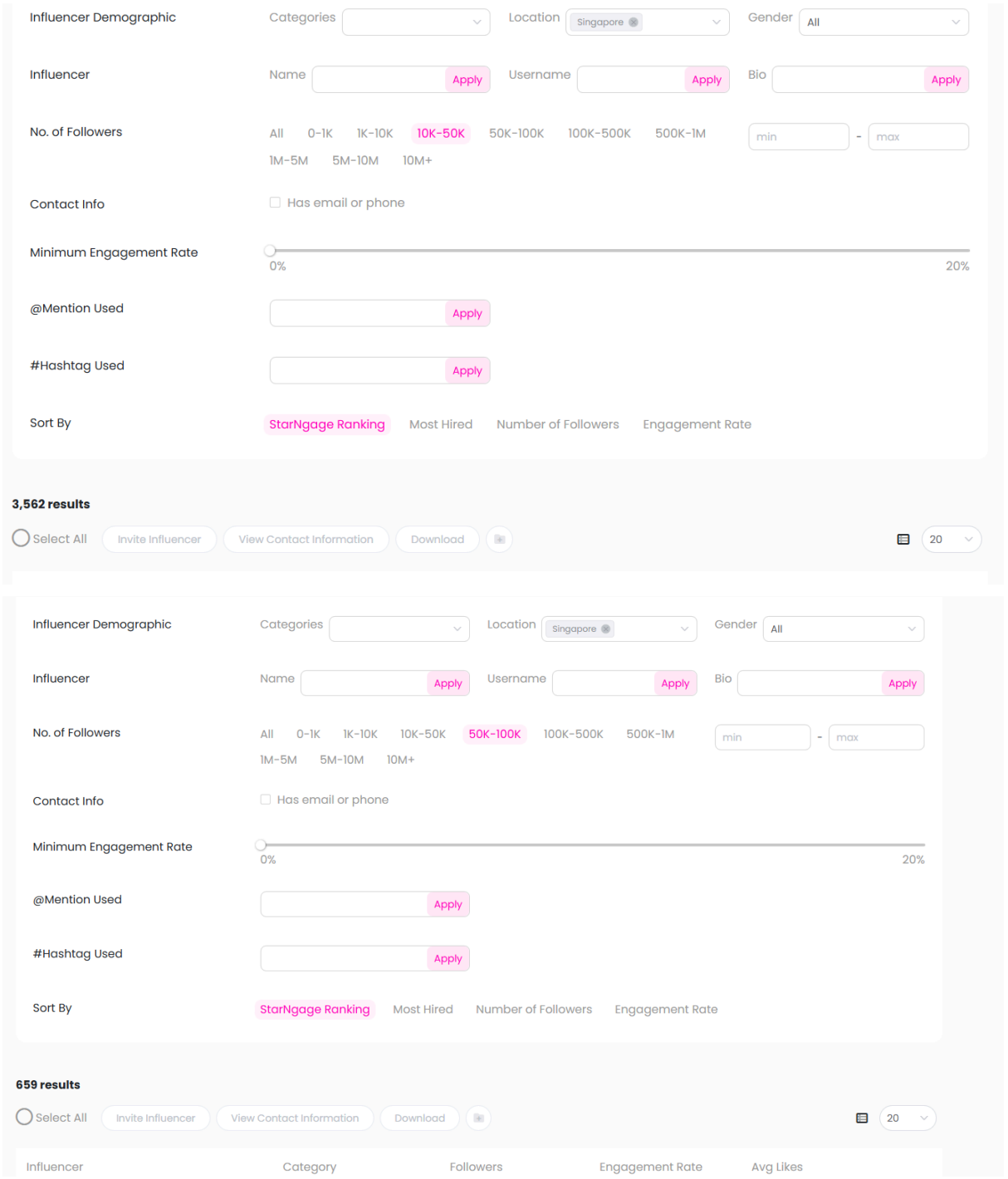
creator-driven campaigns, the demand for specialized tools to manage this content has surged. We identify our Total Addressable Market as this global Influencer Marketing Platform, which was estimated at USD 25.44 billion in 2024. This represents the total potential revenue if our AI solution were to eventually serve the entire global market for creator economy tools.

Serviceable Addressable Market (SAM)
Our immediate focus is on the highly active creator economy in Singapore. Unlike traditional top-down estimates, we have calculated our SAM using a bottom-up approach to ensure accuracy. By leveraging data from regional creator databases such as StarNgage (StarNgage, 2025), we have identified a core target pool of approximately 10,000 nano and micro-influencers (1k-100k followers) currently active in Singapore. Screenshots below are attached to verify the rough numbers that are mentioned here.

Based on a projected annual subscription fee of USD 300 (25 dollar per month) for our AI service, our Serviceable Addressable Market in Singapore is valued at USD 3.0 Million annually. This is generally a modest estimate, there may be even more creators that are not partnered with the database, so we expect this number to be even higher in actuality.

Serviceable Obtainable Market (SOM)
We aim to enter the market by addressing the specific pain points of underserved nano-creators who cannot afford expensive agency tools. Recognizing the competitive landscape, we have set a realistic initial market capture target of 5% of our SAM within the first 18-24 months of operation. This translates to a Serviceable Obtainable Market (an initial annual recurring revenue target) of USD 150,000, representing roughly 500 paid active users. This conservative estimate provides a verifiable baseline for our initial go-to-market strategy before expanding to neighboring SEA regions.


Influencer Demographic
Categories
Location
Singapore
Gender
All
Influencer
Name
Apply
Username
Apply
Bio
Apply
No. of Followers
All
0-1K
1K-10K
10K-50K
50K-100K
100K-500K
500K-1M
1M-5M
5M-10M
10M+
min
max
Contact Info
Has email or phone
Minimum Engagement Rate
0%
20%
@Mention Used
Apply
#Hashtag Used
Apply
Sort By
StarNgage Ranking
Most Hired
Number of Followers
Engagement Rate
7,686 results
Select All
Invite Influencer
View Contact Information
Download
20
Influencer
Category
Followers
Engagement Rate
Avg Likes
Influencer
Name
Apply
Username
Apply
Bio
Apply
No. of Followers
All
0-1K
1K-10K
10K-50K
50K-100K
100K-500K
500K-1M
1M-5M
5M-10M
10M+
min
max
Contact Info
Has email or phone
Minimum Engagement Rate
0%
20%
@Mention Used
Apply
#Hashtag Used
Apply
Sort By
StarNgage Ranking
Most Hired
Number of Followers
Engagement Rate
7,686 results
Select All
Invite Influencer
View Contact Information
Download
20
Influencer
Category
Followers
Engagement Rate
Avg Likes
@Monkey Figure Reviews
Singapore
10K
0.7%
79



Screen shots from StarNgage Platform to size up the micro influencer and nano influencer market in Singapore

Competitor Analysis

To understand the relevance of SocialLens in the market, we compare it with two Singapore-based influencer marketing platforms: Hiip and Narrators. Both competitors primarily serve as intermediaries between brands and creators, helping manage influencer campaigns and measure marketing impact. This comparison highlights the distinct focus and innovation that SocialLens brings to the creator ecosystem.

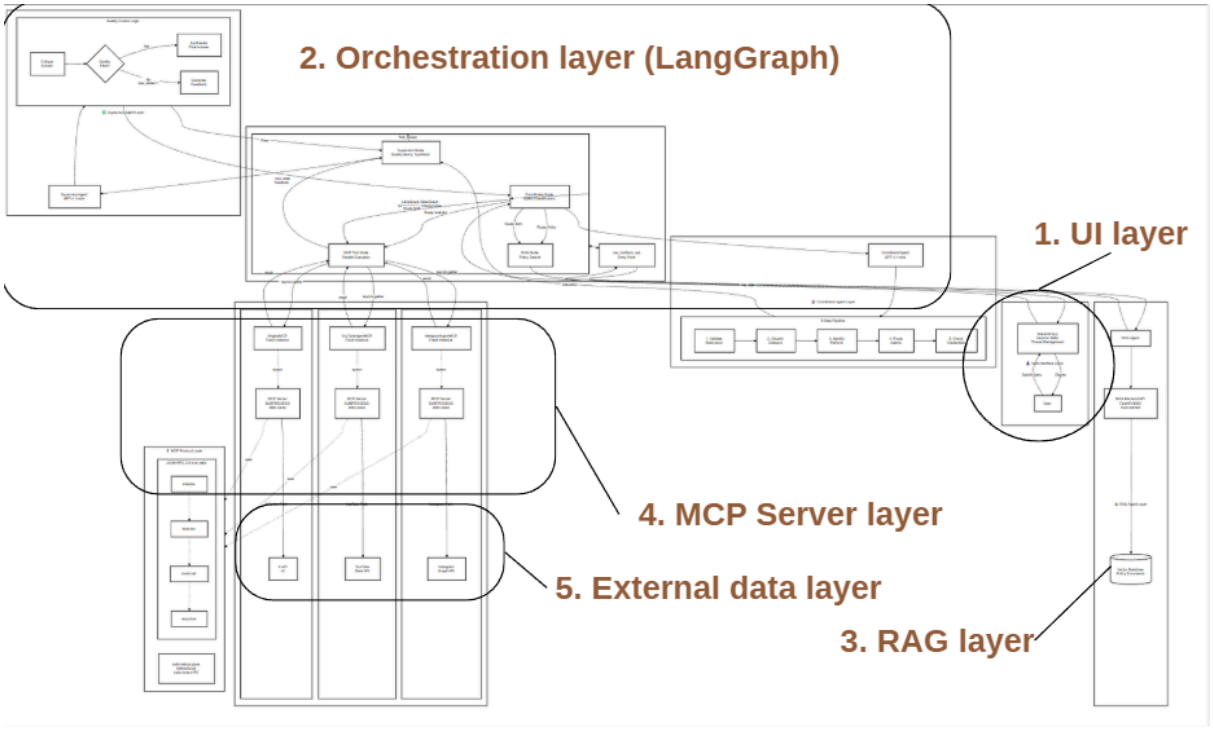
Feature / Capability	Hiip	Narrators	SocialLens

Target Users	Brands and agencies	Brands and content creators (campaign-focused)	Social media influencers and aspiring content creators
AI Usage	Tracks influencer performance, audience sentiment, and engagement	AI-augmented data provides socio-demo, geolocation, audience topics, brand affinity, and campaign performance	Summarizes social media performance, provides personalized content recommendations, and insights across platforms
Compliance / Terms Guidance	Not a focus	Not a focus	Helps creators stay compliant with platform-specific terms
Campaign & Monetization Support	Yes, for brands managing campaigns	Yes, creators earn from brand campaigns	No direct campaign feature. The focus is on creator growth and insights
User Support	Dashboard & analytics	Moderation of content drafts, campaign support, reports with recommendations	Personal assistance, analytics, and compliance feedback

In summary, SocialLens addresses a different set of problems compared to existing influencer marketing platforms. While Hiip and Narrators concentrate on supporting brands through campaign management and influencer performance tracking, SocialLens focuses directly on individual creators. The platform empowers creators of all sizes to gain cross-platform insights, receive data-driven content recommendations, and ensure compliance with platform-specific terms and conditions. By prioritizing creator growth rather than campaign facilitation, SocialLens positions itself as a first-of-its-kind tool for social media creators.

Appendix B: Coordinator Agent

Detailed System Architecture

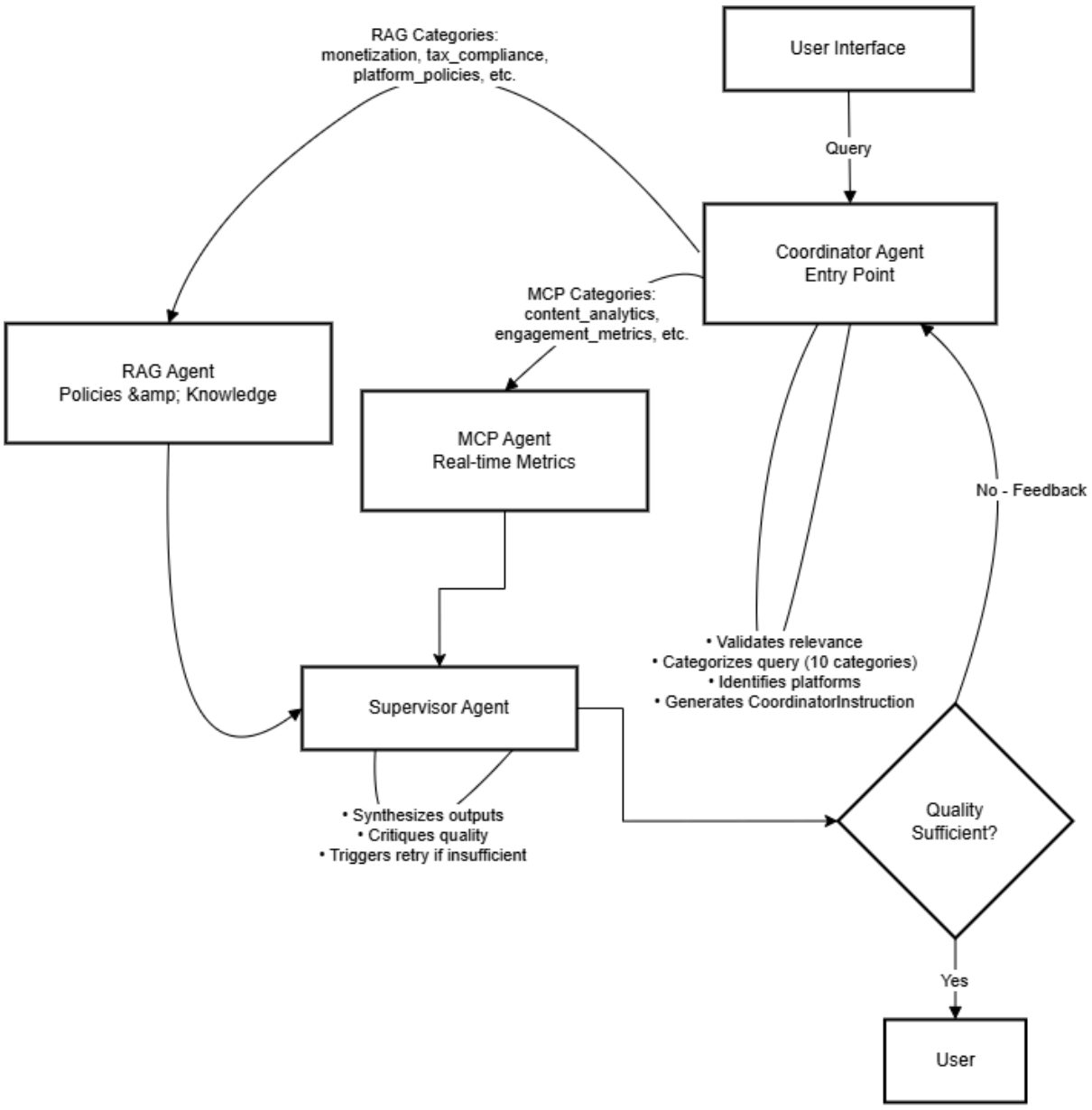


This architecture is structured into five distinct layers, each handling a specific part of the process from user query to final response.

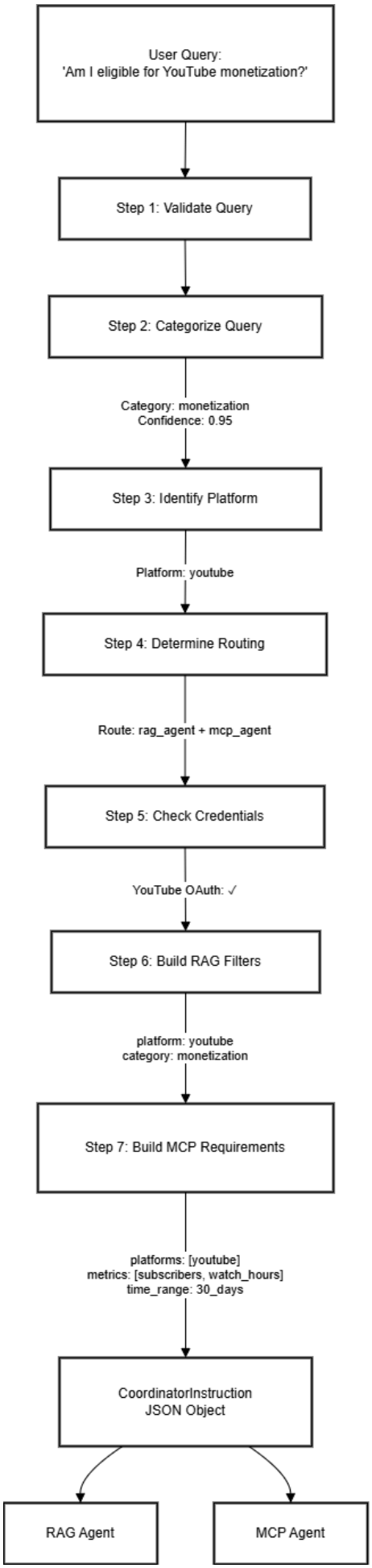
1. 1. UI Layer:
    - This is the user-facing front end.
    - It captures the user's query and displays the final synthesized response.
2. 2. Orchestration Layer (LangGraph):
    - This is the "brain" of the system, built using LangGraph.
    - It receives the query from the UI layer.
    - This layer contains the Coordinator Agent (for routing) and the Supervisor Agent (for quality control and synthesis).
    - It intelligently routes tasks to the RAG layer (for policy questions) or the MCP Server layer (for data/tool use).
3. 3. RAG Layer:
    - This specialized layer is tasked by the Orchestration layer to handle all policy, compliance, and document-related queries.
    - It retrieves relevant information from a vector database (e.g., Supabase).
4. 4. MCP Server Layer:
    - This layer acts as a secure and standardized gateway for platform-specific tools.
    - It hosts the individual MCP servers (e.g., for YouTube, Instagram, X).
    - When a Platform Agent in the Orchestration layer needs data (e.g., analytics), it sends a request to the corresponding MCP server in this layer.
5. 5. External Data Layer:
    - This is the outermost layer, representing the actual third-party APIs (e.g., YouTube Data API, Instagram Data API).
    - The MCP Server layer is the only layer that communicates with this external layer to fetch raw data.

Flow Summary: A user query flows from (1) UI Layer → (2) Orchestration Layer →(3) RAG Layer and/or (4) MCP Server Layer → (5) External Data Layer. The data then flows back to the (2) Orchestration Layer for synthesis before the final answer is sent to the (1) UI Layer.

Architecture diagram + interaction with other agents

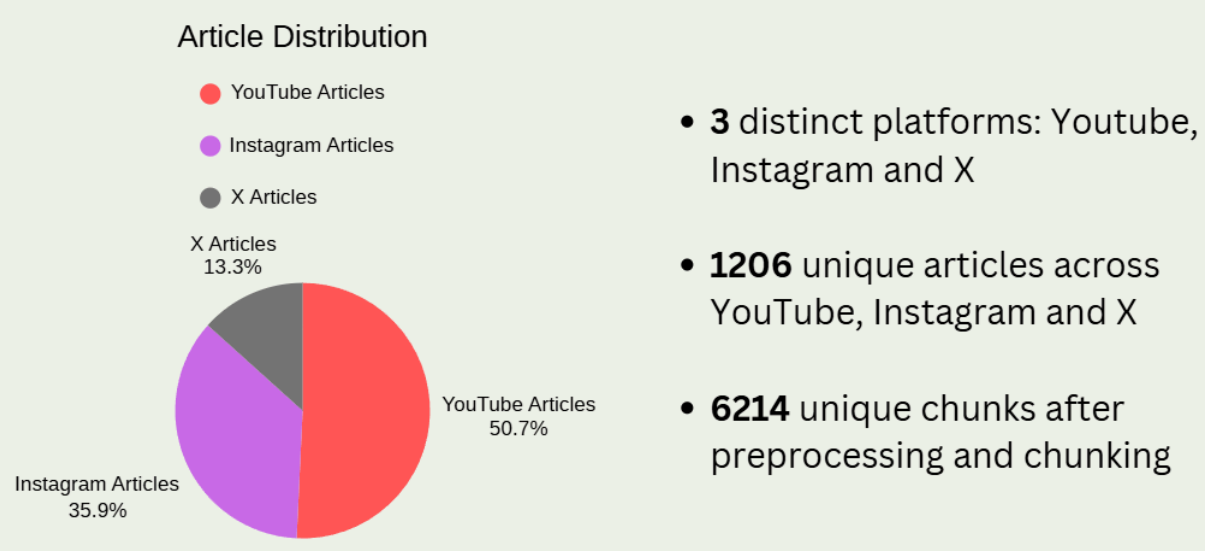


Example data flow



Appendix C: RAG

Current Data Distribution



Technical Choice for Cloud DB

	Supabase	ChromaDB	Weaviate
Native Support	Relational	Vector	Vector
Vector Support	✅ with pgvector extension	✅	✅
Text Seach	✅ with tsvector extension	✅	✅
Serverless/Cloud	✅	✅	✅
Free Tier	✅ Perpetually Free ✅ 500 MB database size ✅ Shared CPU - 500 MB RAM	❌ $5 Free Credit (Limited)	❌ Managed Sandbox Version Free for 14 Days

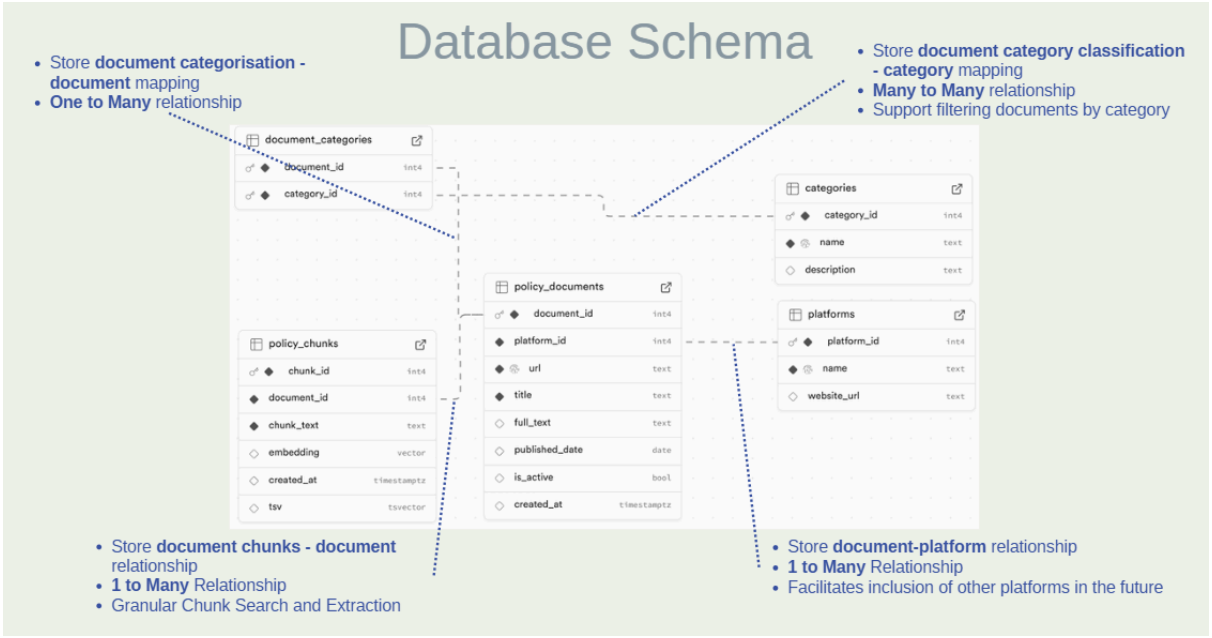
Supavase provided a better free tier cloud database which could facilitate coordination and cooperative work, compared to a locally hosted database which would require setting up on each instance the app is set up.
Postgres + Pgvector extension is also easy to draw on the strengths of relational and vector databases without managing two separate databases.

Database API Endpoints

Method	Endpoint	Description	Request Body (JSON Example)	Response Body (JSON Example)

GET	/health_check	Checks the health of the API, database connection, and embedding model.	(None)	{"status": "ok"} or {"status": "unhealthy", "reason": "..."}
POST	/search	Performs hybrid search (semantic + keyword) with optional filtering by platform and/or category.	{"query": "user question", "top_k": 5, "platform_id": 1, "category_ids": [2, 5]}	[{"chunk_id": 1, "document_id": 1, "content": "...", "source_url": "...", "platform_name": "...", "score": 0.016}, ...]
GET	/platforms	Retrieves the list of all available platforms from the database.	(None)	[{"platform_id": 1, "name": "Platform A"}, {"platform_id": 2, "name": "Platform B"}]
GET	/categories	Retrieves the list of all available categories from the database.	(None)	[{"category_id": 1, "name": "Category A", "description": "..."}, ...]
POST	/find_doc	Retrieves the full document text and metadata for a given document ID.	{"document_id": 123}	[{"document_id": 123, "url": "...", "title": "...", "full_text": "...", "published_date": "..."}]

Database Schema in Supabase



This schema is designed to store and manage text data for a Retrieval-Augmented Generation (RAG) system, enabling efficient, multi-faceted filtering and search.

- **platforms:** This table stores the high-level sources of the documents.
- **policy_documents:** This is the central document metadata table. Each document (primary key document_id) is linked to one platform via platform_id in a **one-to-many relationship**. It also stores the source url and title.
- **policy_chunks:** This table breaks down each document into smaller pieces. Every chunk has a chunk_text, its corresponding vector embedding (for semantic search), and a tsv vector (for keyword search). It links back to its parent policy_documents via document_id in a **one-to-many relationship**.
- **categories:** This table lists all possible classification categories.
- **document_categories:** This is a join table that creates a **many-to-many relationship** between documents and categories. This flexible design allows a single document to be tagged with multiple categories (and a single category to apply to many documents), which is essential for filtering search results.

Hybrid Search

```
WITH base_chunks AS (
    SELECT
        pc.chunk_id,
        pc.document_id,
        pc.chunk_text,
        pc.embedding,
        pc.tsv,
        pd.url AS source_url,
        p.name AS platform_name
    FROM
        policy_chunks pc
    JOIN
        policy_documents pd ON pc.document_id = pd.document_id
    JOIN
        platforms p ON pd.platform_id = p.platform_id
    LEFT JOIN
        document_categories dc ON pd.document_id = dc.document_id
    WHERE
        (%(platform_id)s IS NULL OR pd.platform_id = %(platform_id)s)
    AND
        (%(category_ids)s IS NULL OR dc.category_id = ANY(%(category_ids)s))
    GROUP BY
        pc.chunk_id, pd.document_id, p.name
),
```

Execute Prefiltering on the Platform ID

```
semantic_search AS (
    SELECT
        chunk_id,
        document_id,
        chunk_text,
        source_url,
        platform_name,
        RANK() OVER (ORDER BY embedding <=> %(query_vector)s::vector) as rank
    FROM base_chunks
    ORDER BY embedding <=> %(query_vector)s::vector
    LIMIT 20
),
keyword_search AS (
    SELECT
        chunk_id,
        document_id,
        chunk_text,
        source_url,
        platform_name,
        RANK() OVER (ORDER BY ts_rank_cd(tsv, plainto_tsquery('english', %(query_text)s)) DESC) as rank
    FROM base_chunks
    WHERE tsv @@ plainto_tsquery('english', %(query_text)s)
    ORDER BY ts_rank_cd(tsv, plainto_tsquery('english', %(query_text)s)) DESC
    LIMIT 20
),
combined_results AS (
    SELECT * FROM semantic_search
    UNION ALL
    SELECT * FROM keyword_search
)
```

Execute Semantic search with pgvector native operator (<=>) for cosine similarutym and keyword search using the tsvector column and GIN index.
Extract the top 20 and combine the Results

```
SELECT
    chunk_id,
    document_id,
    chunk_text AS content,
    source_url,
    platform_name,
    SUM(1.0 / (60 + rank)) as rrf_score
FROM combined_results
GROUP BY chunk_id, document_id, chunk_text, source_url, platform_name
ORDER BY rrf_score DESC
LIMIT %(top_k)s;
```

From the candidates score, calculate the RRF with

$$RRF\ score\ (chunk) = \sum_{score \epsilon \{semantic\ score,\ keyword\}} \frac{1}{k + rank(chunk)}$$

The constant 60 is the industry standard for smoothing the drop off between the ranking between chunks (standard ranked reciprocal with no smoothing constant ½, ⅓, ¼ .. creates too big a difference between lower ranked candidates as compared to 1/61, 1/62, 1/63 …)

Technical Consideration for Indexing

Feature	Flat	HNSW	IVF
Use Case	Small datasets where perfect recall is crucial.	Large datasets that require the fastest query times and high accuracy.	Large datasets where a balance between speed and memory is needed.
Speed	Slowest.	Fastest.	Faster than FLAT, but slower than HNSW.
Accuracy	Highest (perfect recall).	High recall.	Good recall, but can have higher variance.
Scalability	Poor Scaling with Large Datasets	Scales well	Scales well.

Currently, our vector search utilizes a "Flat" (exact) index, which is suitable for our initial small dataset where perfect recall is beneficial. However, as we scale our data ingestion and continuously scrape and inject new policies, this brute-force approach will become a performance bottleneck, leading to unacceptable query latency. To address this, we plan to transition to an Approximate Nearest Neighbor (ANN) indexing strategy. We are strongly considering HNSW (Hierarchical Navigable Small Worlds) as it offers the fastest query speeds and scales well with large datasets. A key advantage for our use case is its efficient handling of new data insertions, which avoids the costly, full-index rebuilds often required by IVF Flat, making it ideal for our dynamic and growing knowledge base.

Evaluation Summary (FROM RAG > RAG-evaluation.ipynb)

The test cases were created by using a LLM-as-Author approach to generate diverse, yet realistic questions based on the content of the documents in the database. Test case creation and evaluation methodology is as follows:

1. **Document Selection:** A curated, representative sample of documents across platforms and categories were selected from the database to ensure coverage of various topics
2. **Chunk Extraction:** Each selected document was split into smaller chunks to facilitate targeted question generation, and to specifically identify relevant chunks where question answers can be explicitly found
3. **Question Generation:** A large language model (LLM) was prompted to generate questions based on the content of each document chunk and were tasked to simulate as target audience to ask these questions. The prompts were designed to elicit questions that are relevant to the chunk's content, and create questions that are wholly answerable from the chunk itself (for completeness of evaluation). In this testcase generation, we only consider and create cases where the answers can be found only within the document_chunks (not accounting for multi-chunk or multi-document answers)
4. **Answer Generation:** The LLM was also used to generate the expected answers for each question, based solely on the content of the corresponding document chunk. This ensures that the ground truth answers are accurate and directly tied to the source material.
5. **Validation:** The generated questions and answers were reviewed for quality and relevance, ensuring that they accurately reflect the content of the document chunks and are suitable for evaluating the RAG agent. The questions were also checked to ensure that they were logically coherent, and were realistic queries that a user might ask.

6. **Compilation:** The final set of test cases, including questions, expected answers, and references to the relevant document chunks, were compiled into a structured format (e.g., CSV) for use in the evaluation process.

Interpretation and Explanation of Metrics

For the evaluation, we decided to use industry standard metrics from the RAGAS framework to quantify the performance of the RAG_agent. The key metrics were sourced from this documentation webpage of RAGAS: https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/

Retrieval Metrics:

Document Accuracy: This metric measures the percentage of test cases where the RAG agent successfully retrieved the correct document containing the answer to the query. A higher document accuracy indicates that the agent is effective at identifying and retrieving relevant documents from the database.

- **ID Precision @ k**: This metric indicates the proportion of retrieved document chunks that are relevant to the query. A higher precision means that the RAG agent is effective at filtering out irrelevant information, providing users with more accurate answers.
    - There are multiple ways to compute the precision metrics. There are two broad approaches to computing precision, either by using an LLM or using non-LLM based approaches
    - In this evaluation, we used the ID-based precision metric which relies on assigning unique IDs to each document chunk and comparing the retrieved IDs against the ground truth relevant IDs.
    - id-based precision is calculated using the formula:
    - ID-based Precision = (Number of retrieved context IDs found in reference context IDs) / (Total Number of retrieved context IDs)
    - Since our test cases were designed such that each question has only one relevant chunk and the retrieval searches for the top_k chunks where
    - , the maximum achievable score for precision in this evaluation is 0.2 (1 out of 5 retrieved chunks can be relevant at most) for Precision @ 5. Metrics for precision @ 1 were also calculated to see if the top retrieved chunk was relevant.
- **ID Recall @ k:** This metric measures the proportion of relevant document chunks that were successfully retrieved by the RAG agent. A higher recall indicates that the agent is comprehensive in its retrieval, ensuring that important information is not missed.
    - Likewise, there are multiple ways to compute the recall metrics. There are two broad approaches to computing precision, either by using an LLM or using non-LLM based approaches
    - In this evaluation, we used the ID-based recall metric which relies on assigning unique IDs to each document chunk and comparing the retrieved IDs against the ground truth relevant IDs.
    - id-based precision is calculated using the formula:
    - ID-based Precision = (Number of retrieved context IDs found in reference context IDs) / (Total Number of referenced context IDs)
    - Since our test cases were designed such that each question has only one relevant chunk, the maximum achievable score for precision in this evaluation are either 1 or 0. Metrics for recall @ 1 were also calculated to see if the top retrieved chunk was correct.
- Reciprocal Rank: This metric evaluates the rank position of the first relevant document chunk retrieved by the RAG agent. It is calculated as the inverse of the rank position (1/rank). A higher RR indicates that relevant information is being retrieved earlier in the list, which is crucial for user satisfaction, and helpful for the LLM to generate accurate answers.

Answer Generation Metrics:

- **Faithfulness**: This metric assesses how accurately the RAG agent's generated answers reflect the information contained in the retrieved document chunks. A higher faithfulness score indicates that the agent is effectively utilizing the retrieved information to produce accurate and reliable answers.
    - This is a LLM-as-judge based metric where the LLM is prompted to compare the generated answer against the retrieved contexts to determine if the answer is faithful to the information provided in the contexts.
    - Faithfulness is calculated with the following formula: Faithfulness = (Num claims in the response supported by the retrieved context) / (Total no of claims in the response)
- Response Relevancy: This metric measures the relevance of the RAG agent's generated answers to the original queries. A higher response relevancy score indicates that the agent is providing answers that are directly related to the questions asked, enhancing the overall user experience.
    - The ResponseRelevancy metric measures how relevant a response is to the user input. Higher scores indicate better alignment with the user input, while lower scores are given if the response is incomplete or includes redundant information.
    - Generate a set of artificial questions (default is 3) based on the response. These questions are designed to reflect the content of the response.
    - Compute the cosine similarity between the embedding of the user input and the embedding of each generated question
    - Take the average of these cosine similarity scores to get the Answer Relevancy

Preliminary Evaluation Results

Metric	**Component**	**Aggregated Mean Results**
Document Accuracy	Retrieval	0.957
Precision @ 1	Retrieval	0.653
Precision @5	Retrieval	0.184
Recall @ 1	Retrieval	0.653
Recall @5	Retrieval	0.922
Reciprocal Rank	Retrieval	0.772
Faithfulness	Answer Generation	0.929
Answer Relevancy	Answer Generation	0.917

For precision @5, we are calculating the sum of relevant contexts / total number of contexts retrieved. Since we are evaluating only a single hop query (one ground truth context chunk) and retrieving the top 5 the maximum score is capped at 0.2 using this document ID method of calculating.

Appendix D: MCP

MCP Agent Process Flow / Workflow Diagram

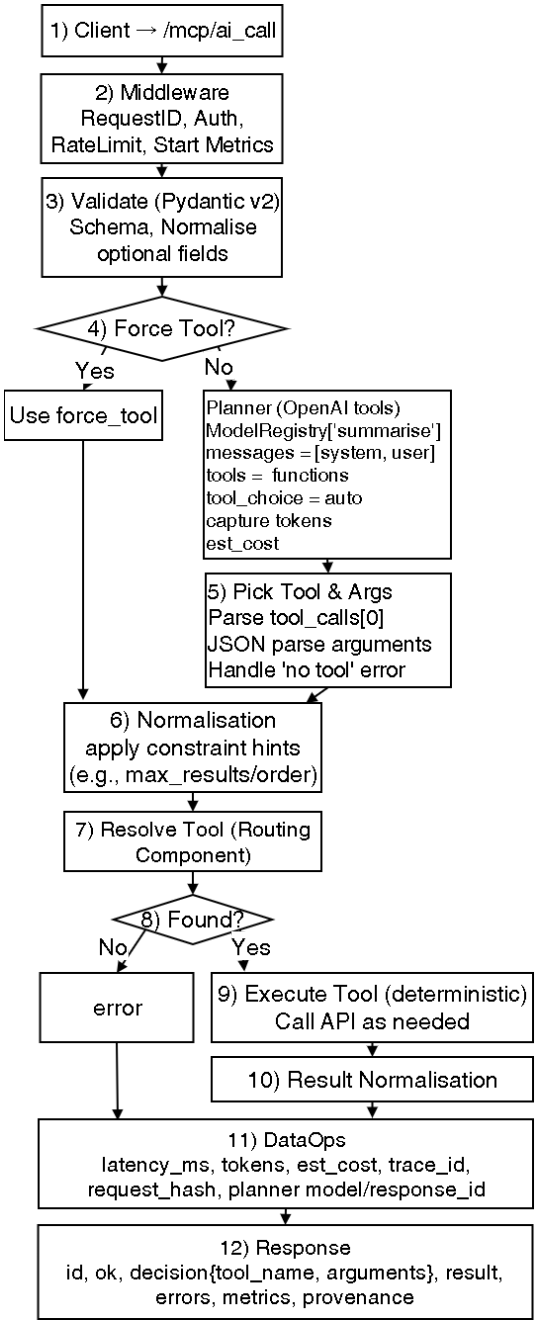


Figure D1: User Process Flow

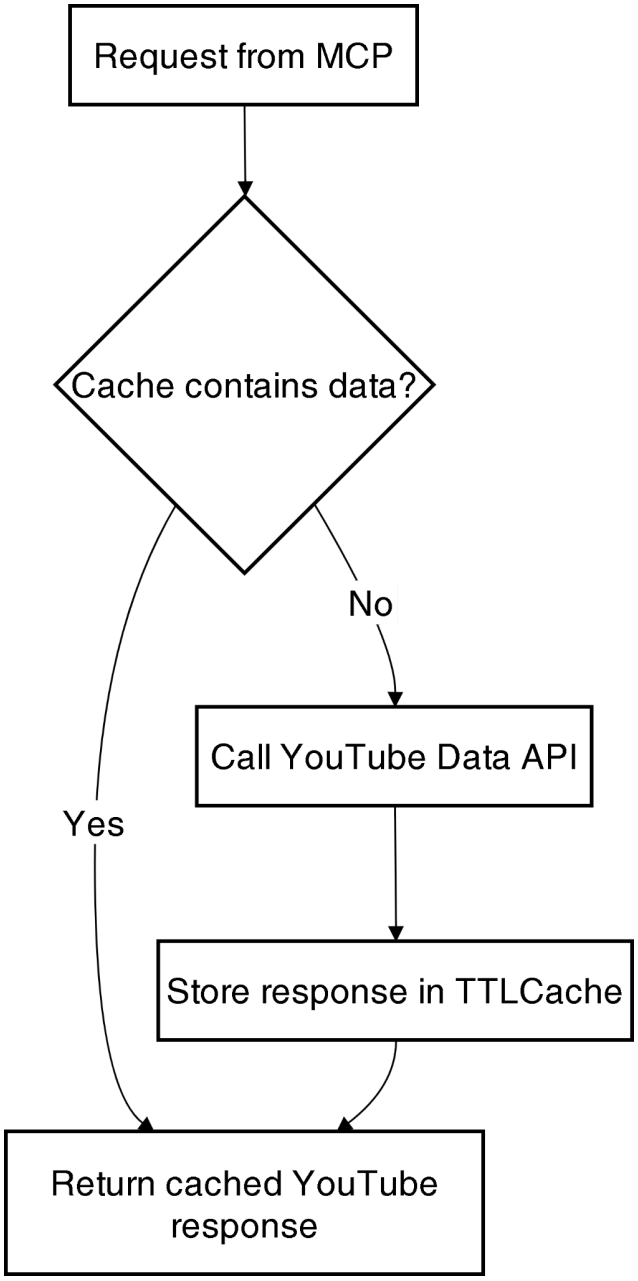


Figure D2: Caching of response

The request enters over HTTPS and passes through a linear middleware chain consisting of RequestID, Auth, RateLimit, MetricsStart, and finally the handler. The RequestID middleware injects an x-request-id into every request and ensures the identifier is propagated back to clients. This matters because it enables log correlation across retries and different client applications, and it is essential for debugging SLA or SLO breaches since engineers can trace a single transaction end to end. Authentication is handled via either an API key or a one-time code. The x-api-key is checked against an allowlist for production integrations, while an alternative x-auth-code (single-use with a time-to-live) supports trials and lightweight evaluations. This design balances business needs by making onboarding easy with codes, securing long-term integrations with keys, and avoiding OAuth complexity for read-only public endpoints. Rate limiting is implemented using a token-bucket per (api-key, ip) with requests-per-second and burst controls. This is an important risk control that prevents accidental thundering herds and enforces fair sharing across creators and tenants. MetricsStart captures both a monotonic start time and wall-clock ISO timestamp so that latency measurement is consistent and cost attribution downstream is accurate. By capturing both time sources, the system avoids clock-skew issues and preserves high-resolution timing needed for performance analysis.

Capabilities and health

The server exposes simple GET endpoints to report health and capabilities. GET /mcp/health returns “ok” when the environment is sane, such as when the OpenAI client is available if that feature is enabled. GET /mcp/capabilities introspects the running service and reports tools from TOOL_IMPLS including alias mappings, schemas derived from Pydantic v2 JSON schema for requests such as MCPCallRequest and YouTube parameter objects, and the current routing table that maps models to tools. Publishing these details has engineering value: it enables programmatic UI generation, serves as a stable contract surface for golden tests, and provides clear self-documentation that reduces ambiguity during development and operations.

Deterministic execution (POST /mcp/call)

The deterministic path begins by validating the payload with Pydantic v2 and clamping fields such as max_results, limit, or top_k to configured caps to stabilise performance and control costs. Idempotency support is provided via an optional Idempotency-Key header and a stable request hash; duplicate requests return a 409 to protect against repeat submissions caused by network errors or retries. Tool resolution proceeds by consulting TOOL_ALIASES and TOOL_IMPLS to obtain the callable function for the requested tool. If the tool is found, the function executes on a worker thread using a non-blocking design so the event loop remains responsive. If the tool is not found, the system can fall back to OpenAI (call_openai) so the LLM provides an answer as a last resort. Results are normalised to plain dictionaries using a _to_dict routine to avoid leaking wrapper or SDK objects into client responses. Metrics include latency and, when OpenAI is used, token counts and estimated cost. Provenance records trace_id, request hash, service and environment, and the OpenAI response id and model when applicable. The server responds with MCPResponse. This approach has several brilliancies: result normalisation keeps client code simple, clamping prevents runaway token and data costs, idempotency guarantees safe retries, and alias-based discovery decouples public names from internal implementations so tools can be refactored freely. The business impact is tangible: influencers receive faster, predictable responses with fewer “try again” hiccups; software and data engineers see fewer edge-case 500 errors, consistent JSON shapes, and deterministic tests; operations benefit from cap enforcement and stable cost curves under load.

Planner (POST /mcp/ai_call)

The planner route accepts AiCallRequest with a natural language query, optional constraints, and an optional force_tool. If force_tool is present, the system skips planning and proceeds directly to deterministic execution; otherwise, it calls OpenAI with a small, inexpensive router-style model using a crisp system prompt and a list of tool schemas for six YouTube functions, with tool_choice set to auto. The planner’s response is parsed to extract tool_calls[0], yielding a tool name and arguments. The system then extracts video IDs from any URLs, resolves the chosen tool, and executes it in the same way as the deterministic path. The result is normalised, planner metrics such as tokens and cost are added, and the server responds with AiCallResponse. This design separates routing costs from execution costs, keeping planning inexpensive while preserving deterministic, typed execution. URL normalisation allows creators to paste raw YouTube links and have the system reliably extract the video_id. Constraints give callers a way to nudge the planner, such as limiting max_results or ordering, without breaking natural-language flows. The business angle is strong: influencers can make natural prompts like “Show analytics for this URL,” engineers can force a specific tool for regression tests and batch jobs, and product teams gain precise cost attribution for planning versus execution and can A/B test routing models without touching tool code.

Tooling model (source of truth) and hot refresh

All business logic resides in plain yt_* functions to maximise testability and performance. If @mcp.tool() wrappers are present, they simply forward calls to the yt_* functions, keeping the wrappers thin. TOOL_IMPLS are discovered at startup, and the administrative endpoint /mcp/admin/refresh_tools supports hot reload of the module and rebuilding of the tool map. This yields engineering value by allowing teams to add or update tools without tearing down the service, which shortens the inner development loop and reduces deployment friction.

Safety, error handling, and resilience

Failures are returned with clear, structured status codes. Authentication failures return 401, rate limits return 429, idempotency duplicates return 409, validation errors return 422, and planner misses return a structured error payload that explains the cause. Errors from OpenAI or Google are captured and surfaced with type and status to aid diagnosis. Timeouts are enforced on external calls to prevent resource starvation and unbounded waits. Every response includes a per-request request-id so clients can correlate logs with server-side traces. The business value is transparency and efficiency: failures are easy to understand, support costs are lower, incident diagnosis is faster, and auditability is maintained.

Observability, metrics, and cost accounting

The system embeds observability into every result by including built-in latency, OpenAI usage, estimated USD cost, and provenance fields. In GCP, Cloud Logging and Monitoring capture per-request details, making it straightforward to build dashboards for cost, performance alarms such as P95 latency, error rates, and RPS, and tenant-level views. A future enhancement is per-tenant labelling for chargeback, which will allow accurate cost allocation and budgeting across customers or internal teams.

MCP Agent Testing/ Error Handling (Unit, Integration, Performance, Security)

Category	Trigger	Status	Message
Security / Auth	Missing/invalid x-api-key or x-auth-code	**401**	{"error":"unauthorised"} or {"error":"invalid_or_expired_code"}
Rate limiting	Token bucket exceeded (key+IP)	**429**	{"error":"rate_limited"}
Input JSON	Malformed JSON body	**400**	{"error":"invalid_json"}
Schema / Validation	Missing fields, wrong types, caps exceeded	**422**	{"error":"validation_error","details":[…]}
Idempotency	Duplicate Idempotency-Key + same hashed body	**409**	{"error":"duplicate_request"}
Routing / Discovery	Unknown tool after alias resolution	**502**	**MCP/AiCall envelope** with errors:[{"type":"unknown_tool"}]
Planner (LLM)	No OPENAI_API_KEY and no force_tool	**400**	{"error":"openai_disabled",...}
Planner selection	LLM returns no tool_calls	**502**	**AiCallResponse** errors:[{"type":"planner"}]
OpenAI API	Model not found, quota, invalid params	**500**	{"error":"internal_server_error",...} **or** MCP envelope with errors:[{"type":"upstream_http"}]

YouTube / Google API	4xx/5xx from Google; OAuth expired	**502**	**MCP/AiCall** errors:[{"type":"tool_error"}] or upstream detail
Internal server	Uncaught exception anywhere	**500**	{"error":"internal_server_error","detail":"…"}
Config / Health	OpenAI enabled but missing key	**500**	plain text openai_misconfig

MCP Agent Evaluation + what could have done to improve given more time (Future Roadmap)

Area	Current posture (observed/assumed)	Risk / cost	Improvement (design & code-level)	Why this helps (justification)	Priority
Routing: NL → single tool	Cheap planner model chooses exactly one tool; args normalised (e.g., URL→video_id)	Planner can pick a real tool name that isn't resolvable at runtime (aliases drift); occasional "no tool_calls"	1) Keep **authoritative TOOL_IMPLS** map + **TOOL_ALIASES** audit on startup and at /mcp/admin/refresh_tools (already added); 2) Add **planner self-check**: force the model to choose from tools[].function.name ∩ TOOL_IMPLS.names and to echo the resolved name; 3) Add **tool-choice confidence**: if no tool, fall back to search_videos with parsed text	Reduces planner misses and "unknown_tool". Tool forcing narrows planner search and improves reliability of function calling. OpenAI function-calling is most robust when you tightly constrain tool names and schemas. (platform.openai.com)	High
Routing: determinism	/mcp/call deterministic; /mcp/ai_call planned then deterministic	Clients might accidentally rely on planner output for repeatability	Expose decision.tool_name and arguments (done) and **allow replay** via /mcp/call using that tuple; add **idempotency** to /mcp/ai_call too	Enables exact replays and golden tests; planner cost minimised on retries; aligns with idempotency guidance. (Stripe)	High
Validation & schema perf	Pydantic v2 models used for requests and capability schemas	Heavy models or custom validators can regress perf	Use **model_validate_json()** where possible; prefer **field validators** over root_validator; keep models flat; avoid excessive generics; enable **orjson** for response serialisation	Pydantic v2's Rust core and model_validate_json can be faster; the 2.x line shows measurable throughput improvements; orjson speeds I/O. (docs.pydantic.dev)	High
HTTP client & I/O	requests for public YouTube calls; googleapiclient (blocking) for OAuth paths; calls run in a worker thread	Extra context switches + per-call DNS/TLS if not pooled	1) Use **httpx** with connection pooling and timeouts for public endpoints; 2) Keep googleapiclient but **batch** where possible (already done for details); 3) Cache discovery docs; 4) Consider **exponential backoff** on 5xx/429	Reduces latency and avoids transient failures; smoother quotas and better tail latency. YouTube has daily quotas and variable per-method costs—avoid wasted requests. (Google for Developers)	High

Concurrency (Cloud Run)	Default concurrency; threaded tool execution	Over-concurrency can cause CPU thrash or memory pressure; under-concurrency wastes CPU	Start with **concurrency ~8**; load test; then raise cautiously. Ensure code is concurrency-safe, and pool sizes match concurrency.	Cloud Run recommends starting at low concurrency and tuning carefully to avoid contention & memory spikes. (Google Cloud Documentation)	High
Caching	In-process TTLCache (per instance)	No cross-instance cache; cache cold after scale-out	Add **Memorystore (Redis)** for shared cache on hot endpoints (search, videos.list); keep per-instance TTLCache as L1	Reduces duplicate hits across instances; improves P95 latency under autoscale; lowers YouTube quota burn. (YouTube quota is limited/day; minimise waste). (Google for Developers)	Medium
Quotas / cost control (YouTube)	Hard caps on max_results; general throttling	Creators can still exceed daily unit budgets during spikes	1) Track **quota units** per endpoint in metrics; 2) Add **tenant budgets** & "soft failover" (degrade to cached) when near limit	YouTube Data API has 10k units/day by default; unit costs differ by method—budget awareness prevents surprises. (Google for Developers)	High
OpenAI cost control	Planner on small model; usage logged	If planner loops or prompt grows, tokens creep	Enforce **token ceilings** on planner; add **fail-closed**: if tool_calls absent, return actionable error and suggestion (or deterministic fallback)	Keeps per-call planner costs predictable; aligns with controlled function-calling practice. (platform.openai.com)	High
Idempotency	Implemented for /mcp/call	/mcp/ai_call not always covered	Add **Idempotency-Key** to /mcp/ai_call; hash (query,constraints); return 409 on dup	Safer retries for planner; avoids duplicate upstream hits. (Stripe)	Medium
Observability	x-request-id, latency, token usage/est.$	Limited model of "what tool" and "YouTube quota"	Emit **tool_name**, **alias_used**, **yt_quota_units**, and **cache_hit** flags; export to Cloud Monitoring	Enables SLOs by tool, and quota dashboards to protect creators' workflows. (Google for Developers)	High
Security	API key / one-time auth code; OAuth tokens	Token on disk (if not moved)	Store tokens in **Secret Manager / GCS**; rotate keys with versions; add **Cloud Armor** rules if public	Standard GCP hardening; reduces blast radius. (Platform docs/guides.) (Google Cloud Documentation)	
Resilience	Generic try/except; upstream exceptions wrapped	No formal circuit breaker	Add **circuit breaker** (short-lived open on repeated 5xx) for OpenAI/YouTube; **exponential backoff** on 429/5xx	Prevents cascading failures and flapping under upstream incidents. (Industry standard pattern.)	

Versioning	Single schema set	Future breaking changes	Add X-API-Version or /v1/… prefix; keep compatibility window	Reduces upgrade friction; allows planner/tool schema iteration safely.	
Extensibility	Hot reload of tools; aliasing	Planner prompt drift vs tool set	Auto-generate **planner tools JSON** from TOOL_IMPLS + TOOL_ALIASES at request time; include descriptions from docstrings	Keeps planner in lock-step with tool registry; lowers maintenance. (platform.openai.com)	
Performance micro-wins	Solid baseline	Extra GC/JSON costs	1) Use **orjson** for responses; 2) Prefer **model_dump(exclude_none=True)**; 3) Precompile regex for YouTube ID (already); 4) Reuse httpx.Client with keep-alive	Shaves ms off hot path; boosts P95/P99 while keeping memory constant. (docs.pydantic.dev)	
Cloud Run scaling	Default autoscaling	Cold starts or thrash on rushes	Set **min instances** (small), **CPU always allocated** if using background I/O; right-size memory (avoid swap thrash)	Improves warm latency without overspend; follow Cloud Run guidance when tuning concurrency. (Google Cloud Documentation)	

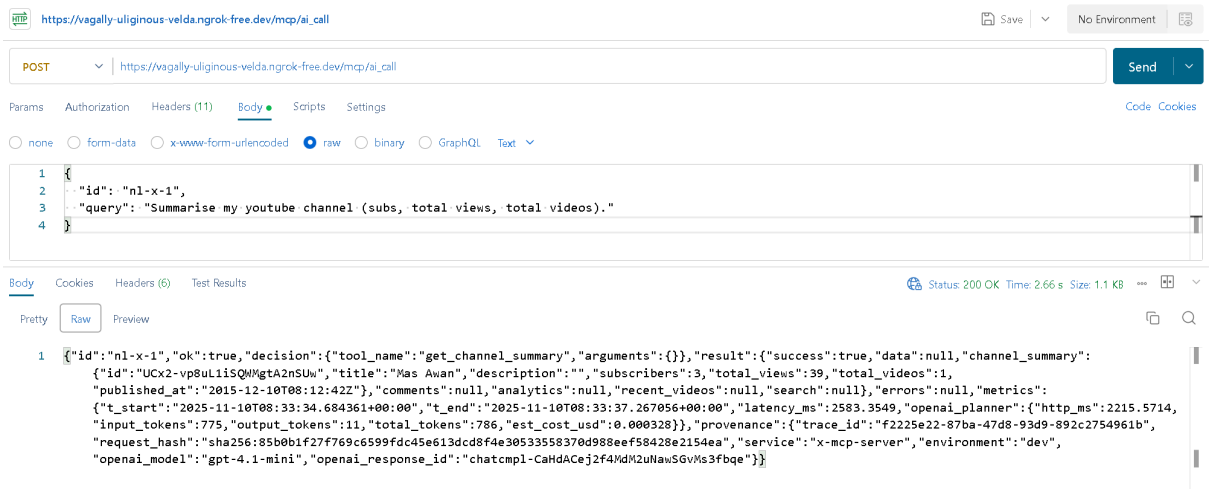
Endpoint	HTTP Method	Purpose
/mcp/health	**GET**	Health check to verify the MCP server is running.
/mcp/capabilities	**GET**	Returns supported tools, models, and features.
/mcp/call	**POST**	Executes a specific tool deterministically (no planner).
/mcp/ai_call	**POST**	Uses the LLM planner to choose and execute the appropriate tool.
/mcp/admin/refresh_tools	**POST**	Hot reloads the tool map for dynamic updates without restarting the server.

MCP Agent Query

```
body = {"id": "nl-search-1", "query": "Search YouTube for 'starlette fastmcp' and show the top 3 results."}
r = requests.post(f"{BASE}/mcp/ai_call", headers=HDRS, data=json.dumps(body), timeout=60)
print(json.dumps(r.json(), indent=2)[:2000])
```


```
{
  "id": "nl-search-1",
  "ok": true,
  "decision": {
    "tool_name": "search_videos",
    "arguments": {
      "query": "starlette fastmcp",
      "max_results": 3
    }
  },
  "result": {
    "success": true,
    "data": null,
    "channel_summary": null,
    "comments": null,
    "analytics": null,
    "recent_videos": null,
    "search": {
      "items": [
        {
          "id": "SvQiSa7ua-Y",
          "title": "What does Starlette really do for FastAPI? - Marcelo Trylesinski",
          "description": "What does Starlette really do for FastAPI? - Marcelo Trylesinski - PyCon Italia 2023 FastAPI has been growing a lot lat
ely, but ...",
          "channel_title": "Python Italia",
          "published_at": "2023-07-31T15:02:35Z"
        },
        {
          "id": "9K0Sols9oD4",
          "title": "Demo on MCP Gateway | OpenAPI + FastMCP + Copilot",
          "description": "Unlock the power of your APIs! In this video, I'll show you how to convert any OpenAPI (Swagger) spec into a usable too
l for AI ...",
          "channel_title": "A-Z Tech Insights",
          "published_at": "2025-05-23T21:57:52Z"
        },
        {
          "id": "Z1XHeMgpg8A",
          "title": "FastAPI in 30 seconds #python #programming #softwareengineer",
          "description": "",
          "channel_title": "Code In a Jiffy",
          "published_at": "2023-02-13T15:57:15Z"
        }

      ]
    }
  },
  "errors": null,
  "metrics": {
    "t_start": "2025-11-02T06:10:56.598556+00:00",
    "t_end": "2025-11-02T06:11:00.069647+00:00",
    "latency_ms": 3469.876,
    "openai_planner": {
      "http_ms": 3061.2561,
      "input_tokens": 410,
      "output_tokens": 225,
      "total_tokens": 635,
      "est_cost_usd": 0.000111
    }
  },
  "provenance": {
    "trace_id": "f19f94d1-69e0-4af4-a041-b343d87756d3",
    "request_hash": "sha256:7494c5e9553eec76b5424e81977c08f1d3a9a396b8ab12f1603fbf0d7af33875",
    "service": "youtube-mcp-server",
    "environment": "dev",
    "openai_mode
```




```
C:\Users\cephe>curl -i https://vagally-uliginous-velda.ngrok-free.dev/mcp/ai_call -X POST -H "x-api-key:devkey123" -d "{\"id\": \"x-1\",\"query\": \"Search For most recent tweets
on apples\"}"
HTTP/1.1 200 OK
Content-Length: 5059
Content-Type: application/json
Date: Mon, 10 Nov 2025 08:20:10 GMT
Ngrok-Agent-Ips: 137.132.26.123
Server: uvicorn
X-Request-Id: c28fc021-547f-4d4d-8ab5-7974e8f51e0d

{"id":"x-1","ok":true,"decision":{"tool_name":"search_tweets","arguments":{"query":"apples","max_results":10}},"result":{"success":true,"data":null,"tweets":{"items":[{"id":"19877
97377755885922","text":"RT @ManCityUS: How do you like them apples? 🍎 https://t.co/KD51L8UhaW","author_id":"1916177160802295808","created_at":"2025-11-10T08:20:01.000Z","lang":"e
n","metrics":{"retweets":211,"replies":0,"likes":0,"quotes":0,"bookmarks":0,"impressions":0},"author_username":"nanakwakye85611","author_name":"nana kwakye"},{"id":"19877973627990
01719","text":"@davido I sell. swap and Buy all kind of Apples products, MacBook,iPad, iWatch,iPhone,Samsungs etc… @Best rate here. \nLocation📍No1 Pepple Street Ikeja Lagos https
://t.co/5DEEabDg16","author_id":"1624767660763365379","created_at":"2025-11-10T08:19:57.000Z","lang":"en","metrics":{"retweets":0,"replies":0,"likes":0,"quotes":0,"bookmarks":0,"i
mpressions":0},"author_username":"Ho_Dreah","author_name":"Ho Dreah"},{"id":"1987797291609071634","text":"@dammiedammie35 I sell. swap and Buy all kind of Apples products, MacBook
,iPad, iWatch,iPhone,Samsungs etc… @Best rate here. \nLocation📍No1 Pepple Street Ikeja Lagos https://t.co/XMQHCclNS","author_id":"1624767660763365379","created_at":"2025-11-10T0
8:19:40.000Z","lang":"en","metrics":{"retweets":0,"replies":0,"likes":0,"quotes":0,"bookmarks":0,"impressions":0},"author_username":"Ho_Dreah","author_name":"Ho Dreah"},{"id":"198
7797246029570451","text":"@DrJJZaato Perhaps we'd appreciate these flawed comparisons (in my opinion) if for the same periods we included the inflation rates, FX rates, and taxes
introduced.\n\nThis oranges to apples comparison put forth by no less a person than you is below par.","author_id":"1191765738555617280","created_at":"2025-11-10T08:19:29.000Z","l
ang":"en","metrics":{"retweets":0,"replies":0,"likes":0,"quotes":0,"bookmarks":0,"impressions":2},"author_username":"RaphaelEyram","author_name":"Eyram"},{"id":"1987797237599019330","text":"@Cleverlydey4u I sell. swap and Buy all kind of Apples products, MacBook,iPad, iWatch,iPhone,Samsungs etc… @Best rate here. \nLocation📍No1 Pepple Street Ikeja Lagos ht
tps://t.co/IY8UL2nKAO","author_id":"1624767660763365379","created_at":"2025-11-10T08:19:27.000Z","lang":"en","metrics":{"retweets":0,"replies":0,"likes":0,"quotes":0,"bookmarks":0
,"impressions":0},"author_username":"Ho_Dreah","author_name":"Ho Dreah"},{"id":"1987797228069765559","text":"RT @taiixyang: He's only 2 apples tall and very nommable \n\nNP WITH
SMOOTO\n#SMOOTOXHENGNAMPINGTRIPD2\n https://t.co/AWqXdmYncs","author_id":"1979926507939319808","created_at":"2025-11-10T08:19:25.000Z","lang":"en","metrics":{"retweets":19,"replie
s":0,"likes":0,"quotes":0,"bookmarks":0,"impressions":0},"author_username":"RuthHe14486","author_name":"he ruth (Ruth)"},{"id":"1987797164337021062","text":"🏋Helped an elderly ma
n carry a bag of apples, kind gesture🍎peppermint tea, cool breath https://t.co/19LH5drMsf","author_id":"3437289494","created_at":"2025-11-10T08:19:10.000Z","lang":"en","metrics":{
"retweets":0,"replies":0,"likes":0,"quotes":0,"bookmarks":0,"impressions":0},"author_username":"pimmtao","author_name":"#PW"},{"id":"1987797148059029740","text":"@_belikebaddy I s
ell. swap and Buy all kind of Apples products, MacBook,iPad, iWatch,iPhone,Samsungs etc… @Best rate here. \nLocation📍No1 Pepple Street Ikeja Lagos https://t.co/BrYkIEy6g5","autho
r_id":"1624767660763365379","created_at":"2025-11-10T08:19:06.000Z","lang":"en","metrics":{"retweets":0,"replies":0,"likes":0,"quotes":0,"bookmarks":0,"impressions":0},"author_use
rname":"Ho_Dreah","author_name":"Ho Dreah"},{"id":"1987797087417688360","text":"@Teeniiola I sell. swap and Buy all kind of Apples products, MacBook,iPad, iWatch,iPhone,Samsungs e
tc… @Best rate here. \nLocation📍No1 Pepple Street Ikeja Lagos https://t.co/8AIhCjoRll","author_id":"1624767660763365379","created_at":"2025-11-10T08:18:51.000Z","lang":"en","metr
ics":{"retweets":0,"replies":0,"likes":0,"quotes":0,"bookmarks":0,"impressions":0},"author_username":"Ho_Dreah","author_name":"Ho Dreah"},{"id":"1987797054098436229","text":"RT @i
luvouy: Apples 🍏\n\n#art #drawing #ocs https://t.co/WrcsaXyAOs","author_id":"1960804168681005056","created_at":"2025-11-10T08:18:43.000Z","lang":"en","metrics":{"retweets":29,"re
plies":0,"likes":0,"quotes":0,"bookmarks":0,"impressions":0},"author_username":"lmimamo","author_name":"Lmima mo haaa???°.·· 🐾 °"}]},"users":null,"created_tweet":null},"errors":nul
l,"metrics":{"t_start":"2025-11-10T08:20:11.275302+00:00","t_end":"2025-11-10T08:20:13.185146+00:00","latency_ms":1909.8664,"openai_planner":{"http_ms":1560.0428,"input_tokens":76
7,"output_tokens":21,"total_tokens":788,"est_cost_usd":0.00034}},"provenance":{"trace_id":"80b7846b-4b83-4590-b42f-7bfc5ef741a2","request_hash":"sha256:5c5da6307a43bb304d969b306c7
331cea75a0094680f3772d517d7eb045a2473","service":"x-mcp-server","environment":"dev","openai_model":"gpt-4.1-mini","openai_response_id":"chatcmpl-CaHQBGJxVaBe5VrwTvhR6bAKbxA4s"}}
C:\Users\cephe>curl -i https://vagally-uliginous-velda.ngrok-free.dev/mcp/ai_call -X POST -H "x-api-key:devkey123" -d "{\"id\": \"x-1\",\"query\": \"Summarise my channel (subs, to
tal views, total videos)\"}"
HTTP/1.1 200 OK
Content-Length: 914
Content-Type: application/json
Date: Mon, 10 Nov 2025 09:59:16 GMT
Ngrok-Agent-Ips: 137.132.212.131
```

Appendix E: Supervisor

Sample Logs of Entire Workflow incorporating all agents.

```
INFO:agents.coordinator_agent: 🔄 Processing query: How can I optimize my Instagram revenue?
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
INFO:agents.coordinator_agent: 📂 Category: revenue_optimization (confidence: 0.95)
INFO:agents.coordinator_agent: 🌐 Platforms: instagram
INFO:agents.coordinator_agent: 🔀 Routing to: rag_agent, mcp_agent
INFO:agents.coordinator_agent: ✅ Instruction generated: coord_fd28ca22

[Coordinator] Query processed → How can I optimize my Instagram revenue?
  → Category: revenue_optimization
  → Platforms: ['instagram']

[RAG] Executing search for: How can I optimize my Instagram revenue?
--- EXECUTING RAG NODE ---
INFO:httpx:HTTP Request: POST http://127.0.0.1:8000/search "HTTP/1.1 200 OK"
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
[RAG] Retrieved 5 documents.
```

```
🔁 Supervisor: still have nodes ['mcp_tool'], returning to coordinator.
INFO:agents.coordinator_agent: 🔄 Processing query: How can I optimize my Instagram revenue?
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
INFO:agents.coordinator_agent: 📂 Category: revenue_optimization (confidence: 0.95)
INFO:agents.coordinator_agent: 🌐 Platforms: instagram
INFO:agents.coordinator_agent: 🔀 Routing to: rag_agent, mcp_agent
INFO:agents.coordinator_agent: ✅ Instruction generated: coord_4e10190e

[Coordinator] Query processed → How can I optimize my Instagram revenue?
  → Category: revenue_optimization
  → Platforms: ['instagram']

[MCP Tool] Processing platforms: ['instagram']
[MCP Tool] Calling instagram agent...
INFO:agents.mcp.instagram_agent: ✅ Instagram MCP session established
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
[MCP Tool] instagram agent completed successfully.

[Supervisor] Combining RAG and MCP outputs.
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"
```

```
[Supervisor] Critiquing the answer (Attempt 1).
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"

[Supervisor] Critique 1: Include specific strategies for advanced monetization methods like Instagram Shopping, Reels bonuses, and leveraging Inst
agram Insights for data-driven decisions to enhance revenue optimization.
Scores → Relevance: 1.0, Completeness: 0.7, Accuracy: 1.0
Confidence Score: 0.90
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"

[Supervisor] Critique 2: Include specific strategies for leveraging Instagram features like Stories, Reels, and Shopping to further optimize reven
ue.
Scores → Relevance: 1.0, Completeness: 0.7, Accuracy: 1.0
Confidence Score: 0.90
INFO:httpx:HTTP Request: POST https://api.openai.com/v1/chat/completions "HTTP/1.1 200 OK"

[Supervisor] Critique 3: Include specific strategies for revenue streams like Instagram Shops, paid subscriptions, and insights on leveraging Inst
agram Reels for monetization to enhance completeness.
Scores → Relevance: 1.0, Completeness: 0.7, Accuracy: 1.0
Confidence Score: 0.90

[Supervisor] Synthesis and critique complete.
Final Answer:
To optimize your Instagram revenue, begin by focusing on growing your follower base and increasing engagement, especially if you are starting from
 zero followers. Consistently post high-quality, valuable content that resonates with your target audience. Utilize relevant hashtags, engage acti
vely with similar accounts, and post during times when your audience is most active to maximize your reach. As your community grows and engagement
 strengthens, you can explore monetization opportunities such as sponsored posts, affiliate marketing, or selling digital products. Prioritizing c
ommunity building and authentic engagement lays a strong foundation for sustainable revenue growth on Instagram.

✅ Supervisor: all work completed. Ending graph.
```

References

Grand View Research. (2024). Influencer Marketing Platform Market Size | Global Industry Report, 2025. Www.grandviewresearch.com. https://www.grandviewresearch.com/industry-analysis/influencer-marketing-platform-market

StarNgage. (2025). #1 Creator Marketplace For Instagram, TikTok & YouTube | StarNgage Plus. Starngage.com. https://starngage.com/plus/en-us

StarNgage. (n.d.). Top Instagram Influencers in Singapore in 2022 | StarNgage. Starngage.com. https://starngage.com/app/global/influencer/ranking/singapore

Yap, H. (2025, April 30). Why it's time we took Singaporean content creators more seriously. Campaign Asia. https://www.campaignasia.com/article/why-its-time-we-took-singaporean-content-creators-more-seriously/502238

Patrecia Meliana. (2025, August). Why mid-tier influencers are being left out. ContentGrip. https://www.contentgrip.com/creator-economy-mid-tier-squeeze/

Ishola, J. (2025, November 4). Why B2B Influencer Marketing Is the New Credibility Currency. Column. https://columncontent.com/b2b-influencer-marketing/

Yeo W.K. (2025) week08-multi_agent_systems [PowerPoint slides]. NUS Canvas. https://canvas.nus.edu.sg/login